Reverse engieering gene regulatory networks: Elucidation of trancriptome organization, gene function and gene regulation in mammalian systems

Thesis

Reverse engieering gene regulatory networks : Elucidation of trancriptome organization, gene function and gene regulation in mammalian systems

Mr Vincenzo Belcastro

The Open University

Year 2010

NO CD/DVD

ATTACHED

PLEASE APPLY

TO

UNIVERSITY

Contents

Date of Submission: 20 September 2010
Date of Award: 22 February 2011

List of Tables

List of Figures

Acknowledgement

...e continuo ad ascoltare Linton Kwesi Johnson, ed a pensarti.

Una e-mail ci ha fatto conoscere, e mi ha fatto scoprire che bella persona che sei. Mi reputo fortunato, abbiamo condiviso molto tempo negli ultimi tre anni, a volte troppo. Diego, mi mancherà lavorare e giocare con te.

Ringrazio Velia per essersi occupata degli esperimenti, e per un po anche di me. Michele Santoro, Giovanni D'Angelo, Nicola e Pratibha per essersi occupati solo degli esperimenti. Francesco Gregoretti per la spensierata collaborazione che ritroverete in forma di codice se avrete la pazienza di leggere i primi 4 capitoli. I ragazzi del gruppo di Bernardo, troppi per essere elencati ma non abbastanza per non ricordarmeli tutti con affetto.

Grazie al Tigem con i suoi Barbara, Marco, Ciro, Agostino, Dina e molte altre allegre persone. Il gruppo bioinformatico con Luisa, Annamaria.

Marika, dovrei ringraziarti per troppe cose...opto per una pizza quando ci vediamo. Ricordo con piacere il tempo trascorso con Donata, Fiorella, Alferio, Cate, Giulia, Nicola, Fabio, Colella, Gopu e gli altri 44 coinquilini conosciuti nella città che mi ha accolto per ben 11 anni, Napoli.

Ringrazio Stefania per le feste in terrazzo, Paola per le esperienze culinarie, Flavio per i mesi di spensieratezza passati insieme. I football warriors per le numerose vittorie a mio favore.

Ringrazio mamma, babbo ed i miei fratelli, mia madre per l'affetto e la pasta al

forno, mio padre per le discussioni ed il vino.

Contributions

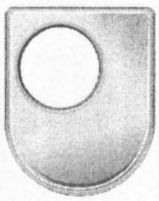
Francesco Gregoretti, *Parallel Implementation of the Mutual Information and Hierarchical Clustering.*

Velia Siciliano, *Yeast-two-Hybrid assays, Biological validation of gene expression localisation. Elucidating the function of the granulin precursor disease-gene.*

Michele Santoro, Giovanni D'Angelo, *Biological validation of gene expression localisation.*

Nicola Brunetti, Pratibha Mithbaokar, *Elucidating the function of the granulin precursor disease-gene.*

Rossella Rispoli, *Identification of Binding sites in Granulin promoter region.*



The Open University

ARC12
(October 2010)

RESEARCH SCHOOL

Affiliated Research Centre Programme

Library Authorisation

Please read and complete this form in conjunction with the ARC\S11G Examination Guidelines for Students. You should return this form to the Research School, The Open University, Walton Hall, Milton Keynes, MK7 6AA with the two bound copies of the thesis and any non-text component, if applicable to be deposited with the University Library. Please note that only theses which comply fully with the binding and presentation criteria as set out in the research degree regulations and the ARC\S11G Examination Guidelines for Students will be accepted for deposition in the University Library. All candidates should complete parts one and two of the form. Part three only applies to PhD candidates.

Part One: Candidate Details

Name: VINCENZO BELCASTRO PI: Y8996002

Degree: PH.D. Affiliated Research Centre: TIGEM

Thesis title: REVERSE ENGINEERING GENE REGULATORY NETWORKS: ELUCIDATION OF TRANSCRIPTOME ORGANISATION, GENE FUNCTION AND GENE REGULATION IN MAMMALIAN SYSTEMS

Part Two: Open University Library Authorisation

I confirm that I am willing for my thesis to be made available to readers by The Open University Library, and that it may be photocopied, subject to the discretion of the Librarian.

Signed: Vincenzo Belcastro Date: 08/04/2011

Part Three: British Library Authorisation [PhD candidates only]

If you want like a copy of your PhD thesis to be available on loan to the British Library Thesis Service as and when it is requested, please tick Section A of this form.

Please note the British Library have requested that theses should be printed on one side only to enable them to produce a clear microfilm. The Open University Library sends the soft bound copy of theses to the British Library.

The University has agreed that your participation in the British Library Thesis Service should be voluntary. Please tick either (a) or (b) to indicate your intentions.

(a) [X] I am willing for The Open University to loan the British Library a copy of my thesis. A signed Agreement Form is attached.

(b) [ ] I do not wish The Open University to loan the British Library a copy of my thesis.

Signed: Vincenzo Belcastro Date: 08/04/2011

An electronic version of this form can be downloaded from http://www.open.ac.uk/research/research-degrees/affiliated-research-centre-programme/affiliated-research-centres.php

Abstract

The main aim of this thesis was to infer mammalian gene regulatory networks from tens of thousands gene expression profiles via a new "reverse-engineering" approach. Human and Mouse gene regulatory networks were both inferred and the results collected in a database that represents part of the non-written material that will support the thesis (http://netview.tigem.it). Each gene regulatory network consists of a set of gene pairs (connections) and a score based on Mutual Information, which states their statistical dependence. The inferred connections are organized into a network that allows exploration of the global features of gene regulation in a mammalian cell. We collected a massive and heterogeneous dataset of 22,255 gene expression profiles from a variety of human samples and experimental conditions. We developed a new mutual-information (MI) reverse-engineering approach able to quantify the extent to which the mRNA levels of two genes are related to each other across such a complex dataset. The resulting network consists of 4,817,629 connections among 22,255 transcripts. The inferred connections were compared against known protein-protein and other regulatory interactions to assess their biological significance. We experimentally identified a subset of predicted protein-protein interactions not previously described in the literature. We discovered regulatory modules within the network, consisting of genes strongly connected to each other, which carry out specific biological functions. We found that these connected genes tend to be in physical proximity at the chromatin level in the nucleus. We show that the network can be used to predict the biological function and subcellular localization of a protein, and to elucidate

the function of a disease-gene. Specifically we discovered that the gene granulin precursor (*GRN*), whose mutations cause frontotemporal lobar degeneration, is involved in lysosome function. We have developed an online tool (http://netview.tigem.it) for querying and exploring the gene regulatory network.

Chapter 1

Introduction

The study of individual gene, and protein function, has been largely the main focus of traditional biology, known as "reductionist" approach. The advent of new technologies, particularly high-throughput technologies, have considerable changed the traditional approach. Nowadays, researchers aim at integrating different sources of information in order to understand the function of a gene as part of a larger network of regulations and interactions.

The common belief that, by identifying one gene responsible for a genetic disease, this would lead quickly to a full understanding of the pathogenesis of the disease, is now known to be wrong. For example proteins have multiple domains (function units) with different "potential" function. So, mutations in one gene would certainly have more than one consequence. This has been confirmed in many species, from baker's yeast to human. So, it is almost impossible to understand the pathogenesis of one disease just by analyzing the function/structure of one gene or its protein product, but the surrounding "gene regulatory network" must also be considered.

Modern biology often follows the *"Holism"*[1] idea, which states that all the properties of a given system cannot be determined by its components parts alone. The general principle of holism was concisely summarized by Aristotle in the *Meta-*

[1]Greek word meaning all, whole, entire, total

physics: "The whole is more than the sum of its parts". Under this paradigm the "systems biology" area of research was born. Systems biology aims to study a biological system as a whole rather than study its components. The main challenge is the identification of "**gene regulatory networks**" by transforming high-throughput datasets into biological insights.

The flow of genetic information has been described as the **central dogma** of molecular biology. The first step is the synthesis of RNA using a DNA-dependent RNA polymerase described as **transcription** schematised with red (DNA) and blue (RNA) blocks in Figure 1.1. The second step is the polypeptide synthesis and is referred as **translation** and schematised with blue blocks (RNA) and circles (proteins) in Figure 1.1. At the level of proteins, among the others, we observe two types of physical interactions: protein-protein interactions, where proteins form a protein complex, and protein-DNA interactions, where "Transcription Factor" proteins bind to the target sequences usually in the immediate vicinity of a gene, and guide the activation of the polymerase that subsequently transcribes the gene. The transcribed gene is referred to as the target of the transcription factor. The process just described is obviously a rough description of the complex regulatory mechanism occurring in a cell. However, for the purposes of the following work, this is a sufficient abstraction of this complex mechanism that is also schematised in Figure 1.1.

In a topological sense, a network is a set of nodes and a set of directed or undirected edges between the nodes. Biological networks exist at various levels as shown in Figure 1.1

- Transcription regulatory networks: Genes are the nodes and edges represent transcription regulation between a transcription factor and a target gene. A gene serves as the source of a direct regulatory edge to a target gene by producing a RNA or protein molecule that functions as a transcriptional activator

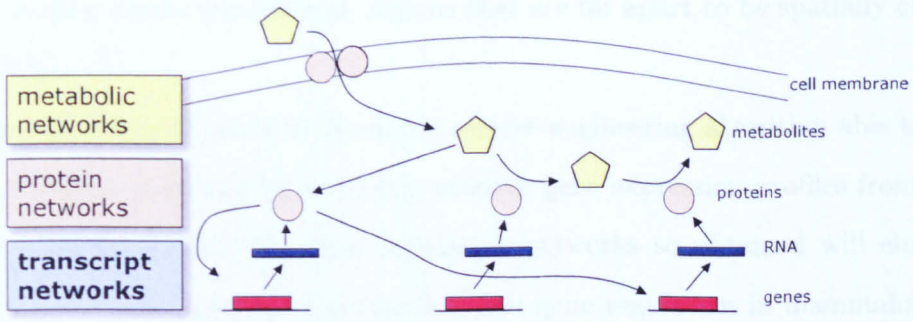


Figure 1.1: Different kinds of networks in Biology.

or inhibitor of the target gene. If the gene is an activator, then it is the source of a positive regulatory connection; if an inhibitor, then it is the source of a negative regulatory connection.

- Protein networks: Proteins are the nodes and edges represent physical interactions between two proteins. These edges are bidirectional (or undirected).

- Metabolite networks: Metabolites are the nodes and an edge represent the reaction catalyzed by an enzyme between the two metabolites. In this kind of network edges are directional.

We must notice that a given interaction (or reaction) can be observed under a certain condition and not in another. Moreover, a gene can be transcribed only under certain conditions. One of these conditions can the tissue. It is known that the genes can be mainly divided in house-keeping (expressed in all the tissues) and tissue-specific. Genes that are only expressed in certain tissues can be inaccessible in others. The accessibility of the DNA sequence representing a gene can be due to the folding of the chromatin structure. Chromosomes in a dividing cell are packed up into neat bundles ready for cell division. In reality, chromosome hence chromatin of the cells during normal activity is folded in such a way that not all the regions can be accessed hence transcribed by the polymerase (heterochromatin). The folding of

the chromatin allows chromosome regions that are far apart to be spatially close to each other.

The aim of this thesis is to develop a reverse-engineering algorithm able to infer gene regulatory networks by analysing massive gene expression profiles from high-troughput technologies. The gene regulatory networks so obtained will elucidate transcriptome organisation, gene function and gene regulation in mammalian systems. The data analysed in this study were downloaded for a public repository for gene expression profiles. The idea is to reverse-engineering mammalians gene networks from a collection of heterogeneous data, hence from different tissues and, in general, different biological conditions that are not uniformly distributed.

The thesis is organized as follows: In Chapter 2, we introduce in more details gene regulatory networks and described various mathematical models used for reverse-engineering the gene regulatory network. In Chapter 3, we performe a comparative study among different approaches to reverse-engineering using ready-to-use software fom each class of models described in Chapter 2. We tested these approaches on experimental data sets as well as on *in-silico* datasets. In Chapter 4, we described a novel approach to reverse-engineering gene regulatory networks using massive datasets of gene expression profiles from human and mouse species. In Chapter 5, we validate the inferred networks by comparing our results with known protein-protein, and other types, of interactions collected from literature. Moreover, we show how to make use of the topology of the gene network to predict gene function. In Chapter 6, we *experimentally* validate some of the new predictions of the reverse-engineering algorithm in terms of new protein-protein interactions and gene function prediction. In Chapter 7, we conclude the thesis highlighting the limitation of the proposed approach and introduced a new statistical model able to generalise the proposed reverse-engineering approach.

The work described in this thesis has been presented in the following publications, [9, 23, 47, 57], and is part of the following manuscripts submitted and in preparation,

[22, 15, 14].

Chapter 2

Introduction to reverse-engineering

Reverse-engineering techniques have principally focused on decoding the mechanisms of transcription control, the first step in gene expression. This is because DNA microarray technology has enabled researchers to efficiently measure the concentration of all RNA transcripts in a cell, making such data abundant. Measuring peptide, protein and metabolite regulators of gene expression is generally more difficult, and such data are not often available. But with improved technologies for protein and metabolite measurement, reverse-engineering techniques may be extended also to these kind of data.

Reverse-engineering techniques can be divided in two main classes: "physical" and "influence" approaches. A physical approach seeks to identify the transcription factors (TF) that regulate gene transcription, and the DNA motifs to which the factors bind. In other words, it seeks to identify true physical interactions between regulatory proteins and their target promoters. An advantage of this strategy is that it can reduce the dimensionality of the reverse-engineering problem by restricting possible regulators to TFs. The second strategy, which we call the "influence" approach, seeks to identify regulatory influences between RNA transcripts. In other

words, it looks for transcripts that act as "inputs", whose concentration changes can explain the changes in "output" transcripts. Each transcript may act as both an input and an output. The input transcripts can be considered the regulators of transcription. By construction, such a model does not generally describe physical interactions between molecules since transcription is rarely controlled directly by RNA (and never by messenger RNA, which is the type of RNA predominantly measured by DNA microarrays). Thus, in general, the regulator transcripts may exert their effect indirectly through the action of proteins, metabolites and effects on the cellular environment (Fig. 2.1). Nevertheless, in some cases, the regulator transcripts may encode the TFs that directly regulate transcription. In such cases, the influence model may accurately reflect a physical interaction. An advantage of the influence strategy is that the model can implicitly capture regulatory mechanisms at the protein and metabolite level that are not experimentally measured. That is, it is not restricted to describing only transcription factor/DNA interactions.

A functional interaction between two genes in a gene network does not necessarily imply a physical interaction, but can also refer to an indirect regulation via proteins, metabolites and ncRNA that have not been measured directly (Figure 2.1). We must mention that the concept of influence interaction is not well defined and strongly depend on the mathematical formalism the gene network is approximated with. Therefore in what follows we will refer to functional interactions by the term "connection", whereas the term "interaction" will be used only when a physical interaction between the DNA, RNA or protein products of the genes is occurring.

The identification of gene regulatory networks is crucial for understanding pathways and functions that take place within a cell. Gene regulatory networks can be inferred by analysing the transcriptional response of a population of cells in multiple experimental conditions. High-throughput technologies such as *microarray* and more recently *next generation sequencing* allow us to measure quantitatively the expression levels of the genes under specific experimental conditions. The capability to

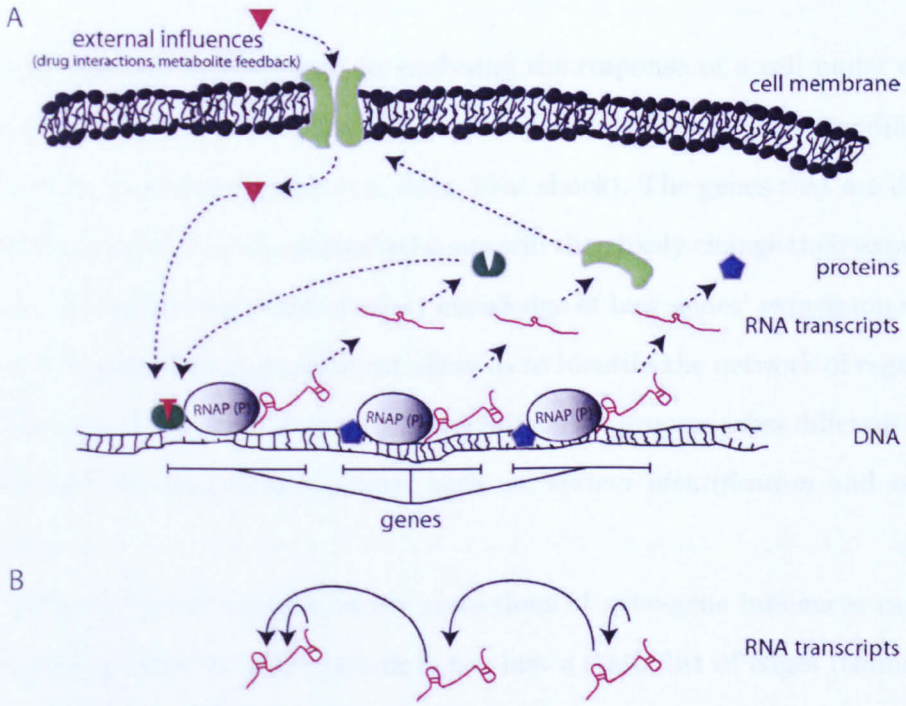


Figure 2.1: Biological networks are regulated at many levels. (A) Shows an example network where transcription factors (blue and green shapes) influence the expression of different transcripts (brown lines). One protein is a membrane-bound metabolite transporter. The metabolite it imports (brown triangle) binds one of the transcription factors enabling it to bind DNA and initiate transcription. (B) A gene network model of the real network in (A). Because the model is inferred from measurements of transcripts only, it describes transcripts as directly influencing the level of each other, even though they do not physically interact.

"obtain" the fingerprint of a cell at a specific time and condition, together with the large number of expression data now available allow to use methods from engineering, mathematics and statistics to explore and analyse gene expression data. In the following chapter, we present different approaches to infer or "reverse-engineer" gene regulatory networks from gene expression profiles measured by microarray technology.

Assume that one is interested in analysing the response of a cell under certain conditions. for example when the expression of some of its genes is modified (or perturbed) by an external agent (i.e. drug, heat shock). The genes that are directly or indirectly regulated by the perturbed genes will then likely change their expression too. From an engineering point of view, knowledge of how genes' expression change following the perturbation experiment allow us to identify the network of regulatory interactions occurring among them. This identification process takes different names depending on the field of application. such as: *system identification* and *reverse-engineering.*

We refer to a *gene network* as the collections of gene-gene influences captured from expression data. A gene network is not just a static list of edges (connections between genes) but it contains information about the topological organisation of its nodes (genes). For example, a community in a network of genes identifies a group of genes that are highly connected among each other and poorly connected with genes outside the group. Communities of genes can be used to detect the modularity of the cell. where groups of genes cooperate to accomplish a specific function.

In what follows we will describe reverse-engineering algorithms to infer gene-gene influence interactions (connections). A description of other methods based on the physical approach and more details on computational aspects can be found in [54, 5, 41, 90, 38, 13, 107].

2.1 Microarray technology and microarray data repositories

A DNA microarray (DNA chip) is a collection of small DNA oligomers on a solid surface of approximately 1 cm^2 (chip). DNA oligomers on the chip are organized in approximately 250,000 "spots" (depending on the chip model), and each spot, called probe, contains millions of copy of the same DNA sequence. Microarrays are used to simultaneously measure the expression of thousands of genes starting from total RNA extracted from a population of cells.

Total RNA is converted into cDNA through reverse transcriptase and marked with a fluorescent marker. cDNA is then placed on the microarray chip and the complementarity between two fragments allows the hybridization of a cDNA sequence to the corresponding DNA "spot". The number of hybridized probes in a spot is directly related to the expression level of the gene represented by the spot. Gene expression levels are quantified through fluorescence analysis, the higher the number of hybridized copy of a probe the higher the fluorescence level measured that is associated to that spot.

There exist many types of microarray all based on the same principles: the two color microarrays are used to measure both the gene expression levels of treated cells and gene expression levels of the control cells on the same chip; other types of microarray can be used to measure single nucleotide polymorphism, fusion genes, alternative splicing, and so on. Here, we concentrate on DNA single color microarray. Hereon we use the term hybridization and gene expression profile (GEP) to refer to the set of gene expression levels collected on a single microarray chip. Moreover, we refer to a set of hybridizations with the term experiment.

The fluorescence levels that are collected from a microarray hybridization are called raw data. The hybridizations in the same experiment are usually "normal-

ized". The normalization process is a data pre-processing step where the measurement noise is removed, the background fluorescence is subtracted and the average fluorescence level among the spots associated to the same probe is computed. The process yields a set of comparable gene expression profiles. We refer to a set of normalized GEPs (or experiment) with the term processed GEPs.

There exist two major GEPs repositories: Gene Expression Omibus (GEO [35]) and Array Express. Array Express [87] is a public repository of gene expression profiles described in literature. GEPs are logically organized into experiments. An experiment is a collection of GEPs (usually performed in a single laboratory) together with their METADATA to trace information such as the applied experimental protocol, type of samples (cell types or tissues), and all the other information required by the MIAME standard [20]. Recently, data from GEO have been curated (METADATA curation) and imported into Array Express in order to be MIAME compliant (data in GEO are not). Most of the experiments in Array Express come with the processed GEPs. Here, we downloaded and analyzed processed GEPs from Array Express repository.

2.2 Reverse-engineering approaches

2.2.1 Bayesian networks

Definition 2.1 *A Bayesian Network (BN) is a directed acyclic graph (DAG)* $G = (V, A)$ *together with a set of local probability distributions* P*. The vertices* V *corresponds to variables, and the arcs or edges* A *represent probabilistic dependency between the variables. An arc from variable* X *to variable* Y *states a probabilistic dependence between the two variables, i.e. the state of* Y *depends on the state of* X*. In this case,* X *is called a parent of* Y*. A node with no parents is unconditional.* P *contains the local probability distributions of each node* X *conditioned on its parents.*

A Bayesian Network provides a graphical representation of probabilistic relationships among a set of random variables X_i, with $i = 1 \ldots n$. An example of Bayesian network is provided in Fig. 2.2. A variable in a Bayesian network can be either discrete or continuous. Bayesian networks can handle incomplete data sets and the statistical nature of the formalism provides a framework to account for noisy data typical of biological experiments.

Owing to its advantages, researchers have devoted considerable attention in recent years to the use of Bayesian network approaches for reverse-engineering gene networks [43, 80, 97, 34, 119, 98, 50, 120, 100, 102].

A Bayesian network describes the relationship between variables at both qualitative and quantitative level. At a qualitative level, the relationships between variables X_i are simply dependence and conditional independence. These relations are encoded in the structure of a directed graph, G, to achieve a compact and interpretable representation. Vertices of the graph correspond to variables and directed edges between vertices represent dependencies between variables.

At a quantitative level, relations between variables are described by a family of joint probability distributions $P(X_1, \ldots, X_n)$ that are consistent with the independence assertions embedded in the graph G and have the form:

$$P(X_1, \ldots, X_N) = \prod_{i=1}^{N} P(X_i = x_i \mid X_j = x_j, \ldots, X_{j+p} = x_{j+p}) \tag{2.1}$$

where the $p + 1$ genes on which the probability is conditioned are called the *parents* of gene i and represent its regulators, and the joint probability density is expressed as a product of conditional probabilities by applying the chain rule of probabilities and independence. This rule is based on Bayes'theorem:

$$P(A, B) = P(B \mid A)P(A) = P(A \mid B)P(B) \tag{2.2}$$

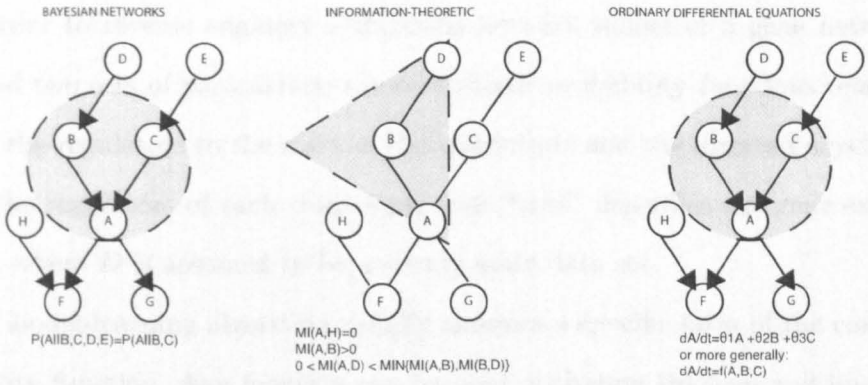


Figure 2.2: Systematic overview of the theory underlying different approaches to reverse-engineering gene regulatory networks. Bayesian networks: A is conditionally independent from D and E given B and C; Information-Theoretic networks: Mutual information is 0 for statistically independent variables, and Data Processing Inequality helps pruning the network; Ordinary Differential Equations: Deterministic approach where the rate of transcription of gene A is a function (f) of the level of its direct causal regulators.

Note that the JPD (joint probabilty distribution) can be decomposed as the product of conditional probabilities as in Eq. 2.1 only if the *Markov assumption* holds, i.e. each variable X_i is independent of its non-descendants, given its parents in the directed acyclic graph G. A schematic overview of the theory underlying Bayesian networks is given in Figure 2.2.

The *Markov assumption* allows to write the conditional probability distribution only conditioning on the parents node. For example in Fig. 2.2 node A is independent from all the others nodes given its parent B and C. The same does not hold if we consider the *Markov blanket* of node A, that states that node A is independent from all the others nodes given its parent, its children and its children's other parents (node H in Figure).

The joint probability distribution, $P(A, \ldots, H)$, for the Bayesian network in Figure 2.2 is given by

$$P(D)P(E)P(H)P(B \mid D)P(C \mid E)P(A \mid B, C)P(F \mid A, H)P(G \mid A) \qquad (2.3)$$

In order to reverse engineer a Bayesian network model of a gene network, we must find two sets of parameters: the conditional probability functions relating the state of the regulators to the state of the transcripts and the directed acyclic graph G (i.e. the regulators of each transcript) that "best" describes the gene expression data D, where D is assumed to be a steady-state data set.

The model-learning algorithm usually assumes a specific form of the conditional probability function. Any function can be used, including Boolean and linear functions. But there will be a tradeoff between model realism and model simplicity. More realistic models will have more parameters, which will require more experimental data and greater computational effort to solve.

The network structure is usually determined using a heuristic search such as greedy-hill climbing approach, stochastic methods or simulated annealing (which do not guarantee convergence to the global optimal solution). Heuristics approaches are used because trying out all possible combinations of interactions among genes is a NP-hard problem. For each network structure G visited in the search, the algorithm learns the maximum likelihood parameters for the conditional probability functions. It then computes a score that evaluates each graph G (i.e. a possible network topology) with respect to the gene expression data D. The score can be defined using Bayes'rule:

$$P(G \mid D) = \frac{P(D \mid G)P(G)}{P(D)} \tag{2.4}$$

where $P(G)$ can either contain some a priori knowledge on network structure, if available, or can be a constant non-informative prior, and $P(D \mid G)$ is a function, to be chosen by the algorithm, that evaluates the probability that the data D has been generated by the graph G. The most popular scores are the Bayesian Information Criteria (BIC) and the Bayesian Dirichlet equivalence (BDe). Both scores incorporate a penalty for complexity to guard against over fitting of data. The BDe score is based on the full Bayesian posterior probability $P(G \mid D)$ and has an inherent

penalty for complexity since it computes the marginal likelihood $P(D \mid G)$ by integrating the probability of the data over all possible parameters assigned to G. The BIC score is an asymptotic approximation to the BDe score that uses an explicitly penalized estimate to the likelihood. One then selects the highest-scoring network as the correct network.

In Bayesian networks, the learning problem is usually underdetermined and several high scoring networks are found. To address this problem, one can use model averaging or bootstrapping to select the most probable regulatory connection and to obtain confidence estimates for the connection. For example, if a particular interaction between two transcripts repeatedly occurs in high-scoring models, one gains confidence that this edge is a true dependency. Alternatively, one can augment an incomplete data set with prior information to help select the most likely model structure.

The main limitation of Bayesian networks is that they assume that the network structure is acyclic (i.e. no feedback loops). *Dynamic Bayesian networks* [119, 34, 80] overcome this limitation and can be used to infer cyclic phenomena such as feedback loops that are prevalent in biological systems. *Dynamic Bayesian networks*, an extension of Bayesian networks, are able to infer interactions from a dataset D consisting of time-series data rather than steady-state data.

A word of caution: Bayesian networks model probabilistic dependencies among variables and not *causality*, i.e. the parents of a node are not necessarily also the direct causes of its behaviour. However, we can interpret the edges as a causal links if we assume that the *Causal Markov Condition* holds. This can be stated simply as: a variable X is independent of every other variable (except the targets of X) conditional on all its direct causes. It is not known whether this assumption is a good approximation of what happens in real biological networks.

For more information and a detailed study of Bayesian networks for gene network inference we refer the reader to [43].

2.2.2 Association Networks

Association networks connect pairs of transcripts that exhibit high statistical similarity (i.e. statistical dependence) in their responses in all experiments in the dataset. To measure similarity, algorithms often use Pearson correlation, which assumes linear dependence between two variables, or mutual information, which makes no assumptions about the form of the dependence. If no assumptions are made, association networks are undirected. Algorithms in this class of model add a connection between all transcripts pairs with expression profiles that exceed a given threshold of similarity. This step, however, does not distinguish between direct and indirect relations. To address this problem, a pruning process can be undertaken to remove connections that are better explained by a more direct path through the graph.

Correlation

Association networks based on correlations often use Pearson correlation coefficient between pair of transcript X and Y to compute similarity. This is computed as:

$$r_{XY} = \frac{\sum_{i=1}^{M}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{M}(x_i - \bar{x})^2 \sum_{k=1}^{M}(y_i - \bar{y})^2}} \tag{2.5}$$

de la Fuente et al [30] used Pearson correlation or Spearman rank correlation to connect all similar transcripts. To prune the network i.e. to remove redundant connections, they used partial correlation coefficients. Partial correlation measures the correlation between two variables after subtracting the correlation between each variable and one or more additional variables.

Mutual Information

Consider a discrete variable X with C possible states, $x_1, \ldots, x_C$ each with its corresponding probability $p(x_i)$. The average amount of information gained from a

measurement that specifies one particular value x_i is given by entropy $H(X)$ and is computed as:

$$H(X) = -\sum_{i=1}^{C} p(x_i) \log(p(x_i)) \tag{2.6}$$

The entropy $H(X)$ has the following properties:

- If an outcome of the measurement is completely determined by x_i i.e. the probability $p(x_i)$ is one and all other probabilities $p(x_j)$ with $i \neq j$ are zero, then $H(X) = 0$.
- For equiprobable events the entropy $H(X)$ is maximum and is given by:

$$p(x_i) = \frac{1}{C} \longrightarrow H(X) = log(C)$$

- Entropy remains unchanged when impossible events are added.
- If the logarithm to base C is used, the entropy is normalized i.e.

$$0 \leq H(X) \leq 1$$

The joint entropy $H(X,Y)$ of two discrete variables X and Y, with Y assuming values in the set $\{y_1, \ldots, y_C\}$, is given by:

$$H(X,Y) = -\sum_{i=1}^{C}\sum_{j=1}^{C} p(x_i, y_j) log(p(x_i, y_j)) \tag{2.7}$$

$p(x_i, y_j)$ denotes the joint probability that X is in state x_i and Y in state y_j. If the variables X and Y are statistically independent the joint probability factorises and the joint entropy $H(X,Y)$ becomes:

$$H(X,Y) = H(X) + H(Y) \tag{2.8}$$

The mutual information $MI(X,Y)$ or MI_{XY} between variables X and Y is defined as:

$$MI(X,Y) = H(X) + H(Y) - H(X,Y) \geq 0 \tag{2.9}$$

Two of the main strategies to estimate MI are:

1. Histogram technique: Consider a collection of N simultaneous measurements of two continuous variables X and Y. First, the range of values of each variable is calculated and then that range is divided into M sub-ranges. The data are thus partitioned into M discrete bins a_i and k_i denotes the number of measurement of X that lie within bin a_i. Probabilities $p(a_i)$ are approximated by the corresponding relative frequencies of occurrence:

$$p(a_i) \rightarrow \frac{k_i}{N}$$

and the $MI(X,Y)$ between two datasets X and Y may be expressed as:

$$MI(X,Y) = logN + \frac{1}{N}\sum_{ij} k_{ij} log\frac{k_{ij}}{k_i k_j} \tag{2.10}$$

k_{ij} denotes the number of measurements where X lies in a_i and Y in b_j.

2. Kernel Density Estimation: With a generalized kernel function $K(X)$, the kernel density estimator $\hat{f}(X)$ is given by:

$$\hat{f}(X) = \frac{1}{Nh}\sum_{i=1}^{N} K\left(\frac{X - x_i}{h}\right) \tag{2.11}$$

The parameter h is called smoothing parameter or window width and the kernel function $K(X)$ is required to be a normalized probability density. For

Gaussian kernel, density estimate then reads:

$$\hat{f}(X) = \frac{1}{N}\frac{1}{h\sqrt{2\pi}}\sum_{i=1}^{N} exp\left(-\frac{(X-x_i)^2}{2h^2}\right) \tag{2.12}$$

For two-dimensional Gaussian kernel estimate is written as:

$$\hat{f}(X,Y) = \frac{1}{2\pi N h^2}\sum_{i=1}^{N} exp\left(-\frac{(X-x_i)^2+(Y-y_i)^2}{2h^2}\right) \tag{2.13}$$

Appropriate value of h depends on the unknown density being estimate. Monte Carlo simulations can be used to obtain optimal value of h [12].

We refer reader to [105] for more details on MI and its estimation.

There are many algorithms which have successfully applied the association network based on MI [21, 12, 39] and shown its application in biological systems.

Relevance Network: Butte et al [21] showed the application of relevance network on 2467 genes in *Saccharomyces cerevisiae*. They estimated the MI pair-wise from 79 measurements using the histogram technique. Each transcript in network was thus completely connected to every other transcript with calculated mutual information. To prune the network, they chose a threshold mutual information (TMI), which was based on estimating average MI after permuting the data n times and selecting the maximum MI. They selected an edge if its MI was higher than TMI.

ARACNe: Basso et al [12] showed the application of ARACNe on 7907 genes in Human B cells. The MI between each pair of genes was estimated using kernel density estimation. To select the smoothing parameter they used Monte Carlo simulation. The algorithm created an initial graph by connecting all transcript pairs with a mutual information value above a p-value computed again using Monte Carlo simulation. Final pruning of the network was done using data processing inequality (DPI) principle that asserts that if both (x, y) and (y, z) are directly interacting and (x, z) is indirectly interacting through y, then $MI(x, z) \leq MI(x, y)$ and

$MI(x,z) \leq MI(y,z)$. ARACNe is more detailed in Chapter 3 in section 3.1.2.

CLR: Faith et al [39] inferred the network of 4345 genes from 445 expression data in *Escherichia coli* using CLR which is an extension of relevance network. For pruning the network, CLR calculated the statistical likelihood of each MI value within its network context by computing the distribution of MI scores for all possible regulators of gene i and distribution of MI scores for all possible targets of gene i.

The definition of MI requires each data point, i.e. each experiment, to be statistically independent from the others. Thus information-theoretic approaches, as described here, can deal with steady-state gene expression data set, or with time-series data as long as the sampling time is long enough to assume that each point is independent.

Edges in networks derived by information-theoretic approaches represent statistical dependences among gene expression profiles. As in the case of Bayesian network, the edge does not represent a direct causal interaction between two genes, but only a statistical dependency.

Theoretically, the main difference between MI and Pearson correlation coefficient is that MI can quantify also *nonlinear* dependencies between variables. Moreover, Pearson correlation cannot imply that two variables are statistically independent. In practical application, however, MI and Pearson correlation may yield almost identical results [105].

In addition MI can be extended to more than two variables, whereas Pearson Correlation is limited to two variables.

2.2.3 Ordinary differential equations (ODEs)

Reverse-engineering algorithms based on ordinary differential equations (ODEs) [17, 44, 112, 118, 111, 108] relate changes in gene transcript concentration to each other and to an external perturbation. By external perturbation we mean an experimental

treatment that can alter the transcription rate of some of the genes in the cell. An example of perturbation is the treatment with a chemical compound (i.e. a drug), or a genetic perturbation involving over-expression or down-regulation of one or more genes.

This is a deterministic approach not based on estimation of conditional probabilities, unlike Bayesian networks and association network approaches. A set of ordinary differential equations, one for each gene, describes the gene regulation as a function of other genes:

$$\dot{x}_i(t) = f_i(x_1, \ldots, x_N, u, \boldsymbol{\theta}_i) \tag{2.14}$$

where $\boldsymbol{\theta}_i$ is a set of parameters describing interactions among genes (the edges of the graph), $i = 1 \ldots N$, $x_i(t)$ is the concentration of transcript i measured at time t, $\dot{x}_i(t)$ is the rate of transcription of transcript i, N is the number of genes, and u is an external perturbation to the system. Since ODEs are deterministic, the interactions among genes ($\boldsymbol{\theta}_i$) represent *causal* interactions, and not statistical dependencies as the other methods.

An algorithm usually presupposes the form of the influence functions f_i and nonlinear functions can lead to exponential rise in the unknown parameters to be estimated. Researchers have studied various functions, including sigmoidal functions [114], linear [32, 44, 26] and non-linear [48] functions.

Reverse-engineering a network using ODEs means choosing a functional form for f_i and then estimating the unknown parameters $\boldsymbol{\theta}_i$ for each i from the gene expression data D using some optimization technique.

The easiest form that this function can assume is the linear form where Equation 2.14 becomes:

$$\dot{x}_i(t) = \sum_j \omega_{ij} x_j + p_i \tag{2.15}$$

where ω_{ij} represents the influence of transcript j on transcript i and p_i is an externally applied perturbation to the level of transcript i. Linear functions have proven

to be the most versatile in the analysis of experimental data sets [32, 44]. In part, this is due to the simplifying power of linear functions; they dramatically reduce the number of parameters needed to describe the influence function and avoid problems with overfitting. Thus, the amount of data required to solve a linear model is much less than that required by more complex nonlinear models. This advantage is crucial in light of the high cost of experimental data and the high dimensionality of the systems. On the other hand, linear functions do not show a rich variety of dynamic behaviour. They only have one isolated stationary state in which the temporal change of transcript vanishes, once reaching this state the concentrations of the network components remain constant. Furthermore, the linear model places a strong constraint on the nature of regulatory interactions in the cell. Therefore, oscillations or multistationarity, which are both important properties of true biological networks, and are nonlinear phenomena, cannot be captured with linear models. Also, higher noise in the microarray data limits their application to make only qualitative statements and not quantitative statements about the underlying network.

ODE-based approaches can be applied to both steady state and time-series expression profiles. The advantage of ODE approaches is that once the parameters, $\boldsymbol{\theta_i}$ for all i are known, Equations 2.14 and 2.15 can be used to make predictions on the behaviour of the network to different conditions (i.e. gene knock-out, treatment with an external agent, etc.) [33].

2.2.4 Other Approaches

Clustering

Clustering, although not properly a network inference algorithm, is the current method of choice to visualize and analyze gene expression data. Clustering is based on the idea of grouping genes with similar expression profiles into clusters [36].

Clusters typically contain genes that function within related pathways or biological processes. It is therefore possible to assign functions to previously uncharacterized genes based on the functions of other genes in the same cluster.

Boolean Networks

Boolean networks [1, 2, 68, 53, 58] offer a binary, discrete-time description of a system. This model assumes that each gene is in two possible states, expressed or not expressed. Interactions among genes are modeled through Boolean logic functions. These models are quite simple and can be easily applied but the underlying assumption seems very unrealistic, in particular, modeling genes as discrete entities assuming one of only two states. Also the amount of data needed to infer the network increases as 2^n, where n is the number of genes.

Chapter 3

Comparison of Reverse-engineering algorithms

In this chapter we test and compare different reverse-engineering algorithms. Each algorithm is based on a different mathematical or statistical model. The input data to the algorithms are the gene expression profiles and, in some cases, details on the performed perturbation experiments. The output is a gene regulatory network where each node is a gene and each edge expresses the influence of one gene on the other. Part of the work described in this chapter has been published in [9].

Generally speaking the identification of a system involves three main entities [73],

1. A collection of experimental data.

   These are usually expression data, which are generated through microarray hybridisation experiments. The data contain the information that allow the identification, or reverse-engineering, of the system that generated them. Gene expression profiles obtained by microarrays are often referred to as observations.

2. A set of candidate models.

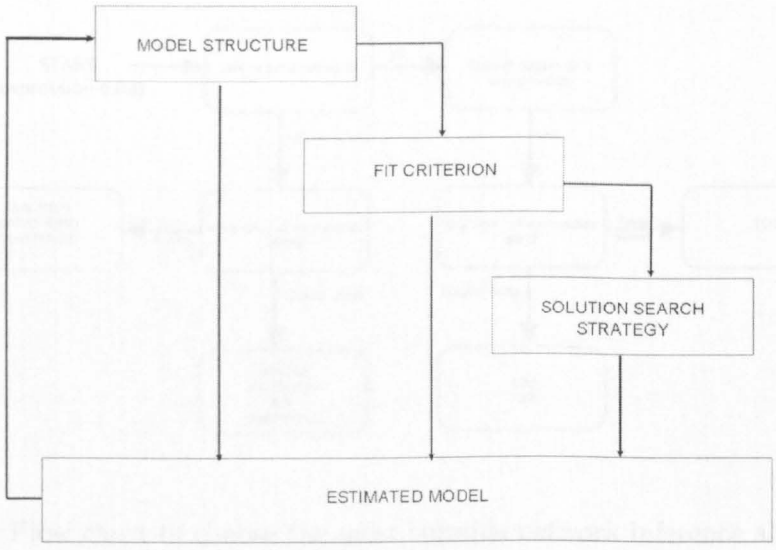


Figure 3.1: System identification process.

The set of candidate models is obtained by specifying the collection of models that can possibly describe the observed data. In this step, it is possible to account for the prior knowledge on the characteristics of the biological system. Each model comes with a set of parameters that can be learned from the data according to how well they explain the observations.

3. A set of rules that allows to accept or reject a candidate model.

   The set of rules is applied over the set of candidate models. The rules are usually expressed in terms of an evaluation function (or objective function) used to check the accuracy of the models in fitting the experimental observations.

Once the model is selected, a model validation step is performed. This step allows to check for the "goodness of fit" of the model to the experimental observations. Figure 3.1 reports the classical system identification process.

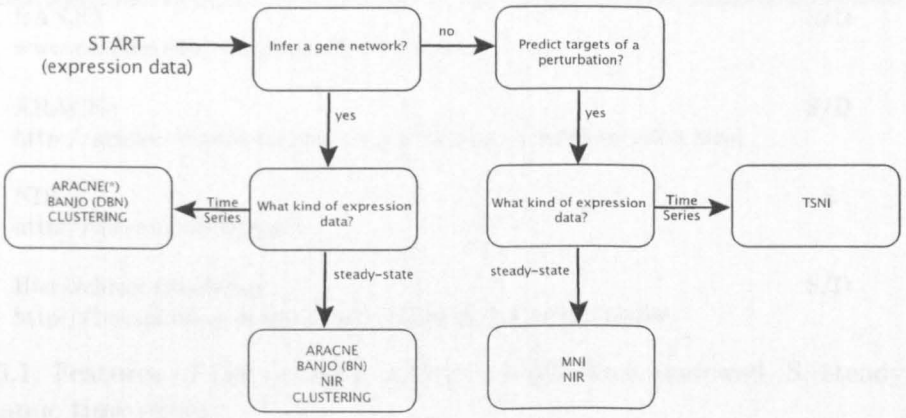


Figure 3.2: Flow chart to choose the most suitable network inference algorithms according to the problem to be addressed. (*): check for independence of time points (see text for details); (BN): Bayesian Networks; (DBN): Dynamic Bayesian Networks.

3.1 Gene network inference algorithms

Hereon we indicate with x_i the set of gene expression measurements of gene i; with D, the set of all the expressions of all the genes, and with a_{ij} a connection between genes i and j, representing a functional or physical interaction between the two genes' products (mRNA or protein).

The choice of the algorithm to reverse-engineer a gene network depends of the type of data available (time-series or steady state). The choice of the algorithm also determines the type of network that is possible to infer (directed or undirected, weighted or not, cyclic or acyclic). An overview of current softwares along with their range of applicability is reported in Figure 3.2 and in Table 3.1, respectively.

3.1.1 BANJO

Reverse-engineering a Bayesian network means to search for the directed acyclic graph, G, that best describes the influence relationships hidden into the data, D, as detailed in Section 2.2.1. Figure 3.3 reports the steps followed by the Banjo [119]

Software	Data type
BANJO www.cs.duke.edu/~amink/software/banjo	S/D
ARACNe http://amdec-bioinfo.cu-genome.org/html/aracneregistration.html	S/D
NIR http://dibernardo.tigem.it	S
Hierarchical Clustering http://bonsai.ims.u-tokyo.ac.jp/ mdehoon/software/cluster/	S/D

Table 3.1: Features of the network inference algorithms reviewed. S: steady-state; D: dynamic time-series.

inference process.

The "searcher" is the core of the Banjo algorithm. The first step, **Proposer**, consists of selecting a graph structure, G_{rough}, to be evaluated according to the data. The strategies currently implemented in Banjo are the *simulated annealing* and the *hill climbing*.

The graph G_{rough} is then scanned for cycles, **Cycle Checker**, and the acyclic graph G generated. The two possible strategies to visit the graph and remove the cycles are the *Breadth First Search* and *Depth First Search*.

The acyclic graph G is then evaluated, **Evaluator**, according to a scoring function. Banjo implements a Bayesian score, defined as the logarithm of the probability of the proposed graph model given the data,

$$BayesianScore(G) = argmax_G \{\log p(G \mid D)\} \tag{3.1}$$

$$= argmax_G \{\log p(G) + \log p(D \mid G) - \log p(D)\}$$

$$\propto argmax_G \{\log p(D \mid G)\} \tag{3.2}$$

where Eq. 3.2 is proportiional to Eq. 3.1 under the hypothesis of equiprobability both over the graph models and over the different data configurations.

The **Decider** decides, possibly stochastically, whether to accept the proposed

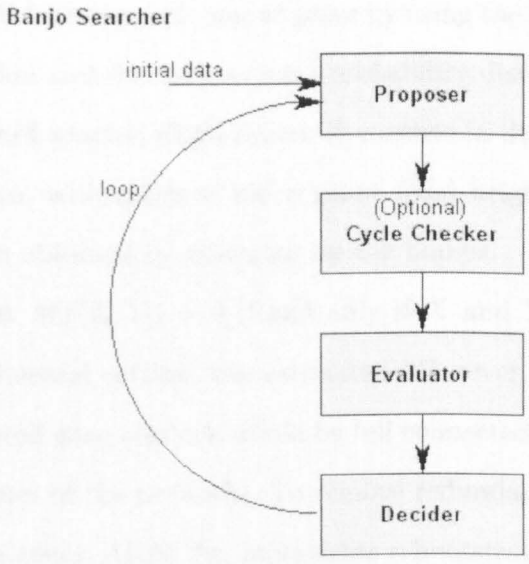


Figure 3.3: Core BANJO Objects

network (as the new current network) and best scored networks are then reported.

3.1.2 ARACNe

ARACNe (Algorithm for the Reconstruction of Accurate Cellular Networks) [12, 76] belongs to the family of association networks for identifying transcriptional interactions between gene products. Relevance networks algorithms, first proposed in [21] (see also [39]), are described in Section 2.2.2.

The computational core of ARACNe consists in the evaluation of pairwise Mutual Information in Equation 2.9 between each pair of genes of interest. The computation of Equation 2.9 requires the knowledge of the marginal and joint probability distributions of the two genes X and Y that have to be estimated from the expression profile. In [16] various methods used to estimate probability distributions from observations are discussed.

ARACNe implements a Gaussian Kernel estimator first reported in [104] that defines a function $\hat{f}$ in terms of a Gaussian kernel function known as *Kernel Estimator*

and estimate the MI between each pair of genes by using the 2.12 as estimation of the marginal distribution and the 2.13 as joint probability distribution. This approach has a straightforward graphic explanation; it consists in drawing M "bumps", one for each observation, with mean x and a priori fixed height. A classical Gaussian shaped plot is than obtained by summing up the bumps.

We remind that $MI(X,Y) = 0$ if and only if X and Y are statistically independent. In experimental setting, the estimated MI never equals zero. Under this scenario the recovered gene network would be full connected (each gene is connected to all the other genes of the network). To remove redundant, hence false predicted connections among genes, ARACNe implements a bootstrap strategy that allows to compute a random MI given the number of observations. This approach allows to set a threshold that discriminates between statistically dependent and independent pairs of genes, given the data.

The threshold over the MIs allows to remove most of the false positives predicted interactions. Biological networks are very sparse and usually the pruning strategy just described is not sufficient. There are cases where, for example, a common regulator of two genes forces the two regulated genes to have an high MI. Consider the network scheme in Figure 2.2, gene B and C will have an high MI score even though they are not directly connected. The information "flows" between B and C through A, hence there is a common shared information (i.e. MI) between B and C, but the interaction is not direct.

The false predicted connections should be kept only when the interest is on the functional relationship between the genes. In this case the recovered network will preserve the connections between the genes whose expression profiles coherently change across a set of hybridisations. ARACNe implements a pruning strategy based on the concept of "Data Processing Inequality" (DPI), in order to remove indirect interactions.

Definition 3.1 *The data processing inequality in information theory states that given three random variables X, Y and Z which form a Markov chain in the order $X \rightarrow Y \rightarrow Z$, then the mutual information between X and Y is greater than or equal to the mutual information between X and Z. That is $MI(X;Y) \geqslant MI(X;Z)$.*

The ARACNe implementation of the DPI scans all the full connected triplets of genes in the network and removes the recovered connection with lowest MI (according to a certain tolerance).

The main limitation of the ARACNe algorithm is that it requires a set of comparable gene expression profiles, i. e. all the expressions it uses to reverse-engineer a gene network must be normalised together. This constitutes a problem in presence of an heterogeneous set of data (different tissues, different laboratories and so on). In Chapter 4 we propose a new reverse-engineering algorithm that overcomes this limitation and thus can be applied to massive and heterogeneous dataset.

3.1.3 NIR

ODE-based algorithms have been developed (Network Identification by multiple Regression, NIR) that use a series of steady state gene expression profiles to reconstruct gene regulatory networks [44].

The network is described as a system of *linear* ordinary differential equations [28] representing the rate of synthesis of a transcript as a function of the concentrations of every other transcript in a cell and the external perturbation:

$$\dot{x}_i(t_k) = \sum_{j=1}^{N} a_{ij} x_j(t_k) + b_i u(t_k), \tag{3.3}$$

where $i = 1 \ldots N; k = 1 \ldots M$, N is the number of genes, M is the number of time-points, $x_i(t_k)$ is the concentration of transcript i measured at time t_k; $\dot{x}_i(t_k)$ is the rate of change of concentration of gene i at time t_k, *i.e.* the first derivative of the

mRNA concentration of gene i measured at time t_k; a_{ij} represents the influence of gene j on gene i; b_i represents the effect of the external perturbation on x_i and $u(t_k)$ represents the external perturbation at time t_k.

In the case of steady-state data, $\dot{x}_i(t_k) = 0$ and Eq. 3.3 for gene i becomes independent of time and can be simplified and rewritten in the form of a linear regression:

$$\sum_{j=1}^{N} a_{ij} x_j = -b_i u \tag{3.4}$$

The NIR algorithm [44] computes the edges a_{ij} from steady-state gene expression data using Eq.(3.4). NIR needs as input: the gene expression profiles following each perturbation experiment (x_j), knowledge of which genes have been directly perturbed in each perturbation experiment ($b_i u$) and optionally, the standard deviation of replicate measurements. NIR is based on a network sparsity assumption, i.e. a maximum number of ingoing edges per gene (maximum number of regulators per gene), which can be chosen by the user. The output is in matrix format where each element is the edge a_{ij}. The inference algorithm reduces to solving Eq.(3.4) for the unknown parameters a_{ij}, i.e. a classic linear regression problem.

Other algorithms based on ODEs have been proposed in literature [17, 111, 32, 108].

The NIR parallel version and its implementation

Equation 3.4 can be rewritten using vector notation:

$$\underline{a}_i^T \underline{x} = -b_i u_i, \; i = 1, \ldots, N. \tag{3.5}$$

Let us suppose that we have conducted M experiments such that we know the genes directly perturbed ($u_i(k)$, $k = 1, \ldots, M$) as well as the expression profiles following the experiments (transcript concentration levels from microarray data $\underline{x}(k)$, $k =$

$1, \ldots, M$). We can then solve the equation (3.5) for the unknown parameters a_{ij}, and thus obtain the ingoing edges for each gene.

NIR applies the multiple linear regression method to estimate the unknown model parameters (a_{ij}). It relies on the assumptions that the data x are realizations of a normally distributed random variable with known variances and the perturbations, u, are perfectly known. Generally, the response y may be related to k regressors and the model

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k, \tag{3.6}$$

is termed a multiple linear regression with k regressors.

Having M experiments (response observation points) at our disposal, then the model becomes:

$$y_i = \beta_0 + \sum_{j=1}^{k} \beta_j x_{ij}, \; i = 1 \ldots M. \tag{3.7}$$

The response ($\underline{y}$) is given by the experimental perturbation values $u_i \in \Re^{1 \times M}$, the regression variable values (X) are given by the concentrations of the gene transcripts and the regression variable parameters are given by the components of the vector $\underline{a}_i$, so that, in matrix form, the model becomes:

$$-\underline{u}_i^T = \underline{a}_i^T X, \; i = 1 \ldots N, \tag{3.8}$$

with $X \in \Re^{N \times M}$. In NIR, the regression analysis aims to best-fit the data by estimating the parameters of the model.

NIR estimates the parameters of the regression variables for each gene, using the least squares method. These are the values for which the first derivative of the residual sum square function is zero:

$$\hat{\underline{a}}_i = -\underline{u}_i X^T (X X^T)^{-1}, \tag{3.9}$$

under the assumption that the regressors are linearly independent.

Biological networks are sparse [83], thus NIR adopts the sparsity assumption that imposes an upper bound on the number of ingoing edges per gene (i.e. maximum number of regulators per gene), $restk < N$, which can be chosen by the user.

For each gene the $restk$ parameters that result in the smallest mean square deviation identify the $restk$ ingoing edges for that gene. The weight of the identified edges is given by the value of the estimated parameters. The choice of $restk$ affects either the sensitivity to measurement errors or the execution time. A low value of $restk$ induces an increase in the solution sensitivity to measurement errors. A high value prohibitively increases the computational time needed to identify the regulatory network due to the high number of the regressor combinations to be included in the model. This number is equal to the number of combinations without repetitions of N objects taken $restk$ at a time:

$$D_{N,restk} = \frac{N!}{restk!(N - restk)!}. \tag{3.10}$$

This is polynomial of degree $restk$ in the number of genes. The exhaustive approach which evaluates the regression for each combination is not feasible for gene networks larger than 100 genes (with 100 genes and $restk = 10$ the number of combinations is of the order of 10^{13}), thus NIR uses the following heuristic approach.

For each gene i:

- At the first step NIR computes (3.9) N times by considering the regression variables one at time; the $topd$ variables for which the sum of the squared deviations is minimized are selected as possible ingoing edges for the gene.

- At the second step NIR computes (3.9) by considering the remaining $N - 1$ variables jointly with each of the first $topd$ selected ones, that is $\frac{topd(2N-topd-1)}{2}$ (Gauss formula) pairs of variables; the $topd$ pairs of variables for which the

sum of the squared deviations is minimized are selected as possible pairs of ingoing edges for the gene.

- At step $k+1$ NIR computes (3.9) by considering the remaining $N-k$ variables jointly with each of the *topd* sets of k variables selected at the previous step, that is $\frac{topd(2N-2k-topd+3)}{2}$ sets of k variables are considered; the *topd* sets of $k+1$ variables for which the sum of the squared deviations is minimized are selected as possible sets of $k+1$ ingoing edges for the gene.

- The process ends when the number of regression variables selected reaches *restk*; the set of *restk* variables for which the sum of the squared deviations is minimized identifies the set of parameters a_i corresponding to the input regulations affecting expression profile of gene i.

The final output is an adjacency matrix, where each element is the edge a_{ij}, that encodes the directed graph. The number of times (3.9) is calculated for each gene is $O(restk \cdot topd \cdot N)$, so the overall number of times (3.9) is calculated is $O(restk \cdot topd \cdot N^2)$. The computational complexity of (3.9) at step $k+1$, for the submatrix of X whose rows correspond to a set of k variables ($0 \leq k \leq restk - 1$), is $O(k^2N)$. The overall computational complexity is therefore $O(topd(restk \cdot N)^3)$.

The NIR algorithm can be easily parallelized to handle large problems in a computationally efficient manner by distributing the overall computational burden among different processors to reduce the total execution time. In order to address the high computational cost issue of the NIR algorithm we have applied some specific implementation optimizations along with parallel programming techniques.

The computational core of the NIR algorithm is the equation (3.9) where X is a submatrix of the gene expression matrix composed only of k rows (with $k = 1, 2, ..., restk$). From the matrix-matrix product definition applied to the submatrix

$X(\underline{v},\underline{v})$, with $\underline{v} = [i_1, \ldots, i_k]$ vector of k indices, it follows that:

$$X(\underline{v},\underline{v})(X(\underline{v},\underline{v}))^T = (XX^T)(\underline{v},\underline{v}). \tag{3.11}$$

Therefore for each step of the algorithm, we don't compute any matrix-matrix product operation $X(\underline{v}, 1 : N)(X(\underline{v}, 1 : N))^T$. On the contrary the product XX^T is computed once and for all at the beginning of the program. At each step k, our implementation just selects the symmetric submatrix of XX^T whose row and column indices correspond to the k possible ingoing edges for the gene. Let S be this submatrix of dimension k stored in packed format.

In each experiment only one gene is perturbed. This implies that for each gene i the perturbation vector $\underline{u}_i$ is equal to $(0, \ldots, 0, \underset{i}{1}, 0, \ldots, 0)$ and then the product $u_i X^T$ reduces to the i-th row of X^T. Denote this row by r.

S is positive definite so we can apply the Cholesky factorization to the matrix S in order to compute $\underline{\hat{a}}_i$ as solution of the system of equations $S\underline{\hat{a}}_i = -r$, thus avoiding the matrix inversion. We rely on the LAPACK routine `DPPSV` to solve this system of equations with a computational complexity of $O(k^3)$.

By avoiding the matrix product in (3.9), the parallel algorithm complexity is decreased by one order of magnitude: at the generic step the computational complexity is $O(k^3)$, the overall computational complexity is therefore $O(topd \cdot restk^4 \cdot N^2)$

The parallelization is implemented by assigning different genes to different computing processes: each process takes care of N/p genes where p is the number of processes available. The computing steps described in the previous section can be performed independently for each gene so each process can compute the results for its genes independently without communication. The parallel algorithm has been implemented in C using the MPI standard.

This work is partly described in [47].

3.2 In-silico and experimental data

Due to the lack of knowledge of all the real interactions between genes in a real biological network, it is very challenging to test gene regulatory network inference algorithms. Moreover, the availability and the heterogeneity of biological experiments do not always allow to reverse-engineer a network of interest. For these reasons we adopted a simulative environment to run the algorithms, test and compare their performances.

3.2.1 Generation of *'In silico'* data

An in-silico gene regulatory network consists of a set of nodes that represent the genes and a set of edges that represent the interactions between the genes. There exist many implementations of network simulators such as [101, 29, 110] able to generate networks whose topology reflects the topology of a biological network (i.e. scale-free and small-world effect, [3, 11, 83]) with different mathematical models for simulating gene expression data.

To simulate gene expression data and gene regulations in the form of a network, we used linear ODEs relating the changes in gene transcript concentration to each other and to the external perturbations (refer to sec. 2.2.3). Linear ODEs can simulate gene networks as directed signed graphs with realistic dynamics and generate both steady-state and time-series gene expression profiles. Linear ODEs are generic, since any non-linear process can be approximated to a linear process, as long as the system is not far from equilibrium, whereas non-linear processes are all different from each other. There are many other choices possible [19] but we valued the capability of linear ODEs of quickly generating many random networks with realistic behaviour and the availability of a general mathematical theory. The code is available in Appendix C.

In-silico networks

The comparative analysis was performed by generating three different sets of 20 networks each, with different network dimensions (10, 100 and 1000) and different average in-degree (2 and 10). To generate each network, we first created a random network using MATLAB© command *rss* that needs as input the order of the model, N, the number of input and output of the system, u and y respectively. The *rss* command generates stable continuous-time state-space models,

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

where x represents the state vector; y is the output vector; u is the input vector; A is the state transition matrix which defines the dynamics of the system; B is the input coefficient matrix: C is the output coefficient matrix; D is the feedthrough (or feedforward) matrix. The *rss* command generates A (N x N), B (N x 1), C (N x N), D (N x 1) which alltogether define the state-space models [72].

The algorithm implemented in the *rss* command, generates a network of size N. It randomly chooses the number of complex and real eigen-values for matrix A (complex eigen-values always exist in pairs, eigen-value and its complex conjugate). For real part of eigen-values, negative random numbers generated from normal distribution are taken and for imaginary part positive random numbers (again from normal distribution). These eigen-values are placed on the diagonal of a zero matrix, Z, of order N x N. If the eigen-value pair is complex, then the real and imaginary part of the eigen-values and its complex conjugate are placed on block diagonal of Z. Another matrix, T, of size N x N is generated randomly from normal distribution and is then orthogonalised. Finally A is computed as $T^{-1}ZT$. B is generated randomly using normal distribution.

Once we generated the state-space model, we set B and C equal to the identity matrix and D equal to a zero matrix so that the output of the system equals its states. Here A represents the network which is a full rank matrix with eigen-values whose real part are less than 0 to ensure the stability of the dynamical system [72] (i.e. all the gene mRNAs reach an equilibrium between their transcription rate and degradation rate after a given time period). For each row of A, we then randomly selected K elements (including the diagonal), and set the other elements to zero (to make the network sparse). We then checked the stability of this new network by computing its eigen-values, and if A was not stable, we then selected a different set of K elements in each row until we got a stable network A.

We used the above-described method to generate networks of size 10 and 100. But to generate network of 1000 genes, we used a different approach. We first generated the network of 100 genes in the same way as described above. We then expanded the network of 100 genes to obtain a network of 1000 genes. To do this, we replicated each row in A, 10 times, adding to each element noise of mean 0 and standard deviation equal to 10% of the absolute value of the corresponding element of the original row in A. While replicating, we removed the diagonal element from the original row and moved it to the diagonal position of the row where it was replicated. It means that each gene has self feedback loop. All the remaining elements in the expanded 1000x1000 matrix were filled with zeros. This is done to simulate clusters of genes that are co-regulated. In the final network of 1000 genes, each gene was controlled only by the original 100 genes in A, and by self feedback, and not from the other genes in the same cluster. We then checked the stability of A by computing its eigen-values that should be negative [72]. If matrix A were not stable, we then repeated the above steps with different random noise until we got a stable network A. In this way, we generated a sparse network in which 100 genes are the main regulators controlling most of the remaining genes in the network, or in network theory words, they act as hubs. Remaining genes are on the network periphery and they do not

control any genes but themselves.

In-silico gene expression data

For each network, we simulated steady state and time-series expression data.

- **Steady State** in which the measurement is taken once the system reaches the steady state after a perturbation.
- **Time Series** in which measurement is taken at M different time points following a perturbation.

Matrix B contains the information of which gene(s) is (are) perturbed in the network. B has all its elements equal to 0 except for the gene(s) that is (are) perturbed. For all datasets, M was chosen equal to 10, 100 and 1000 experiments.

We simulated microarray resulting from two kinds of perturbations, P

1. **Global** perturbation: all the genes are perturbed simultaneously in each perturbation experiment. We randomly generated the perturbation matrix P of size N x M with values generated from Gaussian distribution with mean $\mu = 0$ and standard-deviation $\sigma = 1$. Global perturbations simulate environmental perturbations to the cell state, such as increase of temperature, drug treatment and so on.
2. **Local** perturbation: in each experiment a single different gene is perturbed. We selected P equal to the identity matrix representing single gene perturbation in each experiment. In *local* perturbation, a maximum of $M = N$ experiments can be obtained. Local perturbations simulate, for example, single gene overexpression or knockdown.

Steady State

To simulate the data, we used Eq. 3.4. In compact form it is written as:

$$AX = -BU \tag{3.12}$$

or

$$AX = -P \tag{3.13}$$

where P is equal to BU. Data matrix X is obtained by taking the inverse of A and left multiplying it with the perturbation matrix:

$$X = -A^{-1}P \tag{3.14}$$

Time-series data

Time-series simulated microarray data are generated by perturbing 10% of the genes simultaneously in the network. The genes to be perturbed are chosen randomly and this information is stored in B. U (1 x M) contains the information about what kind of perturbation is applied. In our simulations we applied a constant step perturbation of amplitude equal to 1. Once matrix A (N x N) and B (N x 1) were generated, we simulated the gene expression profile dataset $X = [X(t_1), \ldots, X(t_M)]$ of equally sampled time points which was obtained using *lsim* command in MATLAB© by solving:

$$\dot{X} = AX + BU \tag{3.15}$$

lsim command simulates the time response of the state space model to the given input signal U and gives an output at the sampling time specified. Initial condition for all genes is set to 0. Time t_M was chosen equal to 4 times the inverse of the real part of the smallest eigen-value of A [72]. This ensures that at time t_M all the genes are close to their steady state value.

Dataset	Cell/Organism	Type	Samples	Genes	Reference	True network
A	Human B cells	S	254	7907	[12]	26 MYC targets [12]
B	S. cerevisiae	S	300	6312	[55]	844 TF-gene interactions [67]
C	Human B cells	S	254	23	[12]	11 MYC targets+11 non-targets [12]
D	S. cerevisiae	S	300	90	[55]	subset of TF-gene interactions [67]
E	E. coli	S	9	9	[44]	9 gene network [44]
F	E. coli	T	6	9		9 gene network [44]

Table 3.2: Experimental datasets used as examples. S: steady-state; T: time-series.

Noise in the data

Biological data are noisy. In a real scenario, the inference of a regulatory network is performed on noisy expression profiles. To conform the simulated data to real expressions we added noise to any generated gene expression. The noise follows a Gaussian distribution with zero mean and standard deviation proportional to the level of the expression simulated [44],

$$X_n = X + \sigma_n \mid X \mid \bigodot Y \tag{3.16}$$

where X_n is the noisy data, $\mid X \mid$ represents the absolute values of the elements of X, $\bigodot$ refers to element wise product of two matrices, Y is a matrix of same size as X and is obtained from Gaussian distribution with mean $\mu = 0$ and standard-deviation $\sigma = 1$ (in the simulations $\sigma = 0.1$, 10% of noise).

3.2.2 Experimental data

We selected three different organisms collecting six dataset of gene expression profiles. Table 3.2 lists the organisms selected along with the details of the dataset.

The dataset differ in the number of genes that ranges from tens to thousands. This scenario helps in testing the algorithms in different real experimental settings. The largest dataset, in terms of number of genes, is the *B-cell* dataset (A and C).

As the reference "true network" we used 11 known targets of MYC and 11 that are not targets of MYC [12], to test the predictive performance of different algorithms, on dataset C. For dataset A we used a list of 26 known targets of MYC.

Saccharomyces cerevisiae (dataset B and D) is a well studied organism and many biological confirmed interactions are already catalogued. As the reference "true network" we used 844 TF-target (transcription factor - target) confirmed interactions obtained from Chromatin immunoprecipitatin (ChIP) experiments [67]. As in the case of human B-cells, we also selected a subset of genes to test algorithms on a system of smaller dimension (dataset D).

The *Escherichia coli* dataset contains the expressions of the genes that are involved in the SOS DNA repair pathway of this organism [44]. The steady state expressions, were obtained by over-expressing different genes with an arabinose-controlled episomal expression plasmid. Cells were grown under constant physiological conditions to their steady state (~5.5 hours after the addition of arabinose) and change in expression relative to unperturbed cells was measured. In time-series expressions cells were induced with Norfloxacin drug and total RNA was extracted at the following time points: 0, 12, 24, 36, 48 and 60 minutes from the drug treatment (triplicate[1]).

IRMA: In vivo assesment of Reverse-engineering and Modeling Approaches.

We also tested these reverse-engineering algorithms on a synthetic network (Fig. 3.4) built in *Saccharomyces cerevisiae* [23]. Five well-known genes were chosen in order to obtain a network containing different kinds of interactions, so that it can model both transcriptional and protein-protein interactions, positive and negative feedbacks and it can be turned on and off by changing growing conditions. In order to isolate the synthetic network from the cellular environment non-essential and non-redundant genes were selected and endogenous ones were deleted. The coding sequence of each

[1]http://gardnerlab.bu.edu

gene was assembled with a non-self specific promoter in order to create the desired transcriptional interactions between the network's species, and each cassette was integrated by homologous recombination within the locus of another gene obtaining its simultaneous deletion. Thus, the system was built in such a way that the chosen promoters are regulated only by the transcription factors of the network and not by external gene products. The following genes have been chosen for the synthetic network:

- As activators and repressors encoding genes: CBF1, GAL4, SWI5, GAL80 and ASH1.
- As promoter genes: HO, MET16, GAL1-10 and ASH1.

These activators and repressors regulate each other with the desired network topology, shown in figure 3.4.

Gal4 activates GAL10 promoter only in the presence of galactose that releases the inhibitor Gal80 from Gal4; in contrast when cells are cultured in glucose Gal80 binds to Gal4 activation domain and thus inhibits GAL10 transcription.

The resulting synthetic network is a 'switchable' isolated system which can be easily turned on and off by changing the medium (galactose or glucose). Its behavior can be altered only by perturbing the expression of the network genes. This makes such a network a good model to test the performance of different modeling and inference strategy.

3.2.3 Assessing the performance of algorithms

Suppose to have two graphs with the same set of nodes, how can we define a distance between them? Such a distance could then be used to compare the algorithm performances by setting one of the graph as the true gene network while the other would be the predicted network. We have chosen two parameters to test and compare the

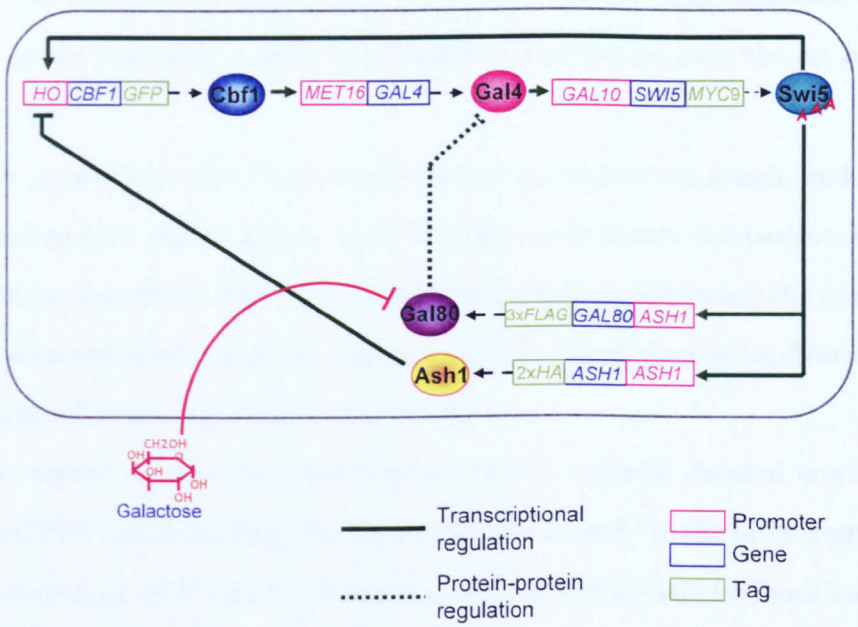


Figure 3.4: Structure of the synthetic network. The genes are all non-essential. Each red box shows the promoter (pr), each blue one the coding sequence of the gene and each green ones the Tag. Solid green lines represent regulation (arrow: upregulation, minus sign: inhibition); dashed black lines represent translation. Ellipses represent protein products. The HO open reading frame was deleted by integrating the CBF1-GFP cassette downstream to HO endogenous promoter. SHE1, CBF1 and SWI5 loci were deleted by integration of MET16pr-GAL4, GAL1-10prSWI5-MYC9 and ASH1prGAL80-3xFlag, respectively. The deletion of SHE2, CBF1 and SWI5 loci has been done from 200-500 bp upstream of the start site to the Stop of each gene. The triple HA tag has been integrated before the Stop codon of ASH1 gene.

predictions: one is the *Positive predicted values* (PPV), that measures the percentage of correct predicted interactions; the other is the *Sensitivity* that measures how much of the real network is covered by the predictions. The two parameters are computed from the *True Positives*, *False Positives* and *False Negatives* predictions as $PPV = TP/(TP + FP)$ and $Sensitivity = TP/(TP + FN)$.

We also consider as random the performance of an algorithm that randomly connect genes assuming a uniform probability distribution over the set of possible edges.

Some algorithms infer the network just as an undirected graph, and others as a directed and/or signed graph, thus, in order to facilitate comparison among algorithms, we computed PPV and Sensitivity by first transforming the real network (signed directed graph) and the inferred network (when directed and/or signed) in an undirected graph (labeled ^{u} in the table).

If the algorithm infers a directed graph and/or a signed directed graph, we also compared PPV and Sensitivity in this case (labeled ^{d} and ^{s} in the table, respectively). While computing PPV and Sensitivity we did not include self-feedback loops (diagonal elements of the adjacency matrix) since all the genes in the simulated networks have self-feedback loops and this could be an advantage for some algorithms as NIR that always recover a network with self-feedbacks.

To transform the signed network to directed network in matrix form, we took the absolute value of the numbers in the weighted adjacency matrix. To transform directed network into undirected network, we symmetrize the matrix and than we select only the values in the upper triangule. While computing PPV and Sensitivity for undirected network, we considered only the upper triangular matrix, i.e. only the upper half of the matrix, A, to avoid counting the connections twice.

To check if the algorithm has performed significantly better than random, we computed the *p-value* using a Binomial distribution to estimate the probability of getting the correct number of edges using a probability of success equal to the

random probability.

Dialogue for Reverse Engineering Assessment and Methods: testing performances

The Dialogue for Reverse Engineering Assessment and Methods (DREAM[1]) [91] is an annual meeting to compare and discuss new reverse-engineering methodologies. It is possible to run an algorithm on biological or simulated dataset, which are provided by the organisers. There are different challenges and one of these consists in reverse-engineering gene regulatory networks from gene expression profiles.

The algorithm performances are compared by applying two scoring metrics: the Area Under the ROC curve (AUROC), which summarizes the tradeoff between the true positive prediction ratio and the false positive prediction ratio; and the Area Under the Precision Recall curve (AUPR) that summarizes the PPV-Sensitivity tradeoff. The predicted edges, sorted according to their significance, may only contain a subset of the true edges. The remaining (null predicted) true edges are then ranked randomly, and this procedure gives rise to a p-value associated to both the AUROC (P_{AUROC}) and the AUPR (P_{AUPR}) scores. The p-values (one for each network), for each metric, are then evaluated together by taking their geometric means. The final *score* is computed as $-\frac{1}{2}\log_{10}(P_{AUROC} \times P_{AUPR})$ [91].

3.3 Results: *'in silico'* evaluation

All of the algorithms were run on all the datasets using default parameters (Appendix A). BANJO was not run on the 1000 genes dataset since it was crashing due to memory limitations, whereas NIR needed an excessively long computation time. The parallel version of NIR [47] overcomes this limitation, as described in Section 3.1.3. Results of the comparison are described in Table 3.3, 3.4 and 3.5.

[1] http://compbio.mit.edu/recombsat/

Datasets	ARACNe PPV/Se	BANJO PPV/Se	NIR PPV/Se	Clustering PPV/Se	Random PPV
10x10	$\mathbf{0.43^u/0.43^u}$	$\mathbf{0.41^u/0.52^u}$	$\mathbf{0.60^u/0.53^u}$	$\mathbf{0.36^u/0.34^u}$	0.36^u
		$0.25^d/0.28^d$	$\mathbf{0.49^d/0.42^d}$		0.20^d
		$\mathbf{0.19^s/0.06^s}$	$\mathbf{0.47^s/0.40^s}$		0.10^s
10x100	$\mathbf{0.63^u/0.65^u}$	$\mathbf{0.97^u/0.09^u}$	$\mathbf{0.72^u/0.88^u}$	$0.40^u/0.38^u$	0.36^u
		$\mathbf{0.78^d/0.07^d}$	$\mathbf{0.69^d/0.88^d}$		0.20^d
		$\mathbf{0.80^s/0.06^s}$	$\mathbf{0.69^s/0.88^s}$		0.10^s
100x10	$0.21^u/0.12^u$	$0.20^u/0.04^u$	$\mathbf{0.26^u/0.04^u}$	$0.20^u/0.12^u$	0.19^u
		$0.10^d/0.02^d$	$\mathbf{0.19^d/0.03^d}$		0.10^d
		$0.07^s/0.01^s$	$\mathbf{0.17^s/0.02^s}$		0.05^s
100x100	$\mathbf{0.29^u/0.24^u}$	$\mathbf{0.71^u/0.00^u}$	$\mathbf{0.72^u/0.63^u}$	$\mathbf{0.24^u/0.14^u}$	0.19^u
		$\mathbf{0.48^d/0.00^d}$	$\mathbf{0.70^d/0.62^d}$		0.10^d
		$\mathbf{0.74^s/0.00^s}$	$\mathbf{0.70^s/0.62^s}$		0.05^s
100x1000	$\mathbf{0.57^u/0.44^u}$	$\mathbf{0.99^u/0.05^u}$	$\mathbf{0.93^u/0.84^u}$	$\mathbf{0.27^u/0.17^u}$	0.19^u
		$\mathbf{0.65^d/0.03^d}$	$\mathbf{0.92^d/0.84^d}$		0.10^d
		$\mathbf{0.66^s/0.03^s}$	$\mathbf{0.92^s/0.84^s}$		0.05^s
1000x1000	$\mathbf{0.04^u/0.23^u}$	-	-	$\mathbf{0.06^u/0.03^u}$	0.02^u

Table 3.3: Results of the application of network inference algorithms on the simulated Global perturbation. PPV: Positive Predicted Value; Se: Sensitivity. In bold the algorithms that perform significantly better than random, using as a random model a Binomial distribution.

Datasets	ARACNe PPV/Se	BANJO PPV/Se	NIR PPV/Se	Clustering PPV/Se	Random PPV
10x10	**0.53^u/0.61^u**	0.41^u/0.50^u	**0.63^u/0.96^u**	0.39^u/0.38^u	0.36^u
		0.25^d/0.18^d	**0.57^d/0.93^d**		0.20^d
		0.15^s/0.05^s	**0.57^s/0.93^s**		0.10^s
100x100	**0.56^u/0.28^u**	**0.71^u/0.00^u**	**0.97^u/0.87^u**	**0.29^u/0.18^u**	0.19^u
		0.42^d/0.00^d	**0.96^d/0.86^d**		0.10^d
		0.60^s/0.00^s	**0.96^s/0.86^s**		0.05^s
1000x1000	**0.66^u/0.65^u**	-	**0.91^u/0.82^u(*)**	**0.20^u/0.10^u**	0.02^u
		-	**0.96^d/0.86^d(*)**		0.01^u

Table 3.4: Results of the application of network inference algorithms on the simulated Local perturbation. PPV: Positive Predicted Value; Se: Sensitivity. In bold the algorithms that perform significantly better than random, using as a random model a Binomial distribution. (*) Results obtained running the parallel version of NIR [47]

Algorithm performances are higher when applied on local perturbations data (one gene perturbed at time), with respect to global perturbation data (multiple genes perturbed). Multiple perturbation data should carry more information with respect to single perturbation data due to the higher number of genes perturbed. However, results show that algorithms are able to recover better a signal from single perturbation data. Also the number of expression profiles is an important parameter. When we try to recover a network by using very few hybridisation with respect to the number of genes, due to the weakness of the signal, performances are very low independently from the algorithm applied (only NIR is slightly better than random).

Despite the dataset configuration BANJO's sensitivity (or network coverage) is always very low with a very high PPV, meaning that despite the very few predicted connections these are indeed correct. For the remaining algorithms, performances improve when using local perturbation data (Table 3.4).

Performance is again random for the time-series 'dynamic' dataset (Table 3.5), except for BANJO albeit with a very low Sensitivity. In this case, we ran ARACNe as well, although the time-points cannot be assumed independent from each other. BANJO has been shown to work on dynamic data but needs a very high number of

Datasets	ARACNe PPV/Se	BANJO PPV/Se	NIR PPV/Se	Clustering PPV/Se	Random PPV
10x10	$0.35^u/0.84^u$	$0.36^u/0.34^u$ $0.24^d/0.22^d$ $\mathbf{0.39^s/0.01^s}$	-	$0.34^u/0.35^u$	0.36^u 0.20^d 0.10^s
10x100	$0.36^u/0.96^u$	$0.38^u/0.45^u$ $0.23^d/0.29^d$ $\mathbf{0.20^s/0.13^s}$	-	$0.33^u/0.34^u$	0.36^u 0.20^d 0.10^s
100x10	$0.19^u/0.63^u$	$0.18^u/0.05^u$ $0.10^d/0.03^d$ $0.03^s/0.00^s$	-	$0.19^u/0.27^u$	0.19^u 0.10^d 0.05^s
100x100	$0.19^u/0.84^u$	$0.11^u/0.03^u$ $0.11^d/0.03^d$ $0.06^s/0.01^s$	-	$0.19^u/0.31^u$	0.19^u 0.10^d 0.05^s
100x1000	$0.19^u/0.87^u$	$0.20^u/0.03^u$ $0.11^d/0.02^d$ $0.06^s/0.01^s$	-	$0.19^u/0.32^u$	0.19^u 0.10^d 0.05^s
1000x1000	$0.02^u/0.96^u$	-	-	$0.02^u/0.37^u$	0.02^u

Table 3.5: Results of the application of network inference algorithms on the simulated Time Series data. PPV: Positive Predicted Value; Se: Sensitivity. In bold the algorithms that perform significantly better than random, using as a random model a Binomial distribution.

number of procs	time (secs)
1	98896
10	9725
20	4798
40	2406
60	1643
80	1259
100	969

Table 3.6: Total execution times in seconds

experiments (time-points) as compared to the number of genes [119].

3.3.1 Application of parallel NIR

In order to measure the result accuracy we ran the program, by using the local steady-state 'in-silico' data discussed in Section 3.2.1, on 20 different networks counting 1000 genes, with 10 as average in-degree per gene.

The program has been executed on 100 processors of an HP XC6000 Cluster with Itanium 2 biprocessors nodes and a Quadrics ELAN 4 network. On average it took 984 seconds to generate the results for each gene network. The program recovered most of the true interactions as shown in the Table 3.4.

Parameters were: $restk = 11$, $topd = 50$. As shown in Table 3.4, NIR performances were better than the one of ARACNe, even in the case of 1000 gene networks.

Moreover we ran the program on a 2500 gene network. We set the same values for the parameters *restk* and *topd* as before. It took around 12,450 seconds to generate the results and we obtained the following values for PPV and Sensitivity: 0.26^d, 0.27^u and 0.10^d, 0.11^u (random PPV: 0.008^u and 0.004^s), respectively. In order to measure the parallel efficiency we run the program for the same 1000 gene network on different number of processors. The execution times are shown in Table 3.6.

Datasets	ARACNe PPV/Se	BANJO PPV/Se	NIR PPV/Se	Clustering PPV/Se	Random PPV
A	$\mathbf{0.14^u/0.35^u}$	-	-	$0.02^u/1.00^u$	0.00^u
B	$0.00^u/0.01^u$	-	-	$0.00^u/0.21^u$	0.00^u
C	$\mathbf{0.78^u/0.64^u}$	$0.60^u/0.27^u$	-	$0.45^u/0.91^u$	0.48^u
D	$\mathbf{0.07^u/0.17^u}$	$\mathbf{0.17^u/0.02^u}$	-	$\mathbf{0.11^u/0.44^u}$	0.02^u
E	$0.69^u/0.34^u$	$0.78^u/0.44^u$ $0.67^d/0.24^d$ $0.50^s/0.02^s$	$\mathbf{0.80^u/0.88^u}$ $\mathbf{0.74^d/0.67^d}$ $\mathbf{0.59^s/0.53^s}$	$0.8^u/0.63^u$	0.71^u 0.63^d 0.32^s
F	$0.75^u/0.37^u$	$0.73^u/0.69^u$ $0.61^d/0.39^d$ $0.00^s/0.00^s$	-	$\mathbf{0.90^u/0.59^u}$	0.71^u 0.63^d 0.32^s

Table 3.7: Results of the application of network inference algorithms on the experimental datasets. PPV: Positive Predicted Value; Se: Sensitivity. In bold the algorithms that perform significantly better than random, using as a random model a Binomial distribution. Details concerning the datasets analysed are reported in Table 3.2.

3.4 Results: Experimental evaluation

The algorithms have been tested on real expression data collected from literature. The details of the data are reported in Table 3.2. The results of the reverse-engineering is reported in Table 3.7. We were unable to run all the algorithms on all the dataset because of the limitations outlined in Section 3.3. However, we should point out that these results could not be suitable for algorithm comparison purposes due to the limited dataset dimensions and to the limited knowledge of the real biological networks.

Due to the dimensions of the dataset, ARACNe was the only software able to infer the network from dataset A and B. Dataset E and F are very small and come from dense biological networks forcing the random performance to be very high. NIR can only be applied on dataset E, because all the other datasets were missing the information regarding the target of the perturbation.

3.4.1 Reverse Engineering the IRMA network

In order to reverse-engineer the IRMA network, we used gene expressions collected after having grown IRMA on two different medium, one containing glucose (switch-off) and one containing galactose (switch-on). To generate **steady-state** data we perturbed this network of 5 genes by over-expressing each gene in a separate experiment (one gene at time). To over-express the genes, we expressed them under the control of a strong constitutive GPD promoter.

Time-series data were collected by growing IRMA cells both in glucose (switch-off) and galactose (switch-on). Measurements were taken every 20 minutes by using real time PCR over the transcript of the 5 genes.

Application of the Banjo algorithm

In order to reverse-engineer IRMA we applied Banjo to the "switch on" and "switch off" time-series data. Banjo recovers the Dynamic Bayesian Network that better describes the observed data. In order to estimate the joint probability distribution (see Section 2.2.1) of all the variables in the network, Banjo first discretizes the data using a *quantile discretization procedure*. The *Proposer/Sercher* strategies were set to *random local move* and *simulated annealing*, respectively. The amount of time Banjo uses to explore the Bayesian Network space was set to *one minute*. All the other parameters such as *reannealingTemperature*, *coolingFactor*, and so on, were left with their default values. Of course the parameter values were not arbitrary chosen; those values were selected as the best values (in terms of network inference accuracy) in [9]. As well as for time-series observations, Banjo is able to infer the Directed Acyclic Bayesian Network on steady state observations. We applied Banjo on both "Glucose steady-state", "Galactose steady-state" dataset and to the switch on and switch off time-series. Results are reported in Figure 3.5.

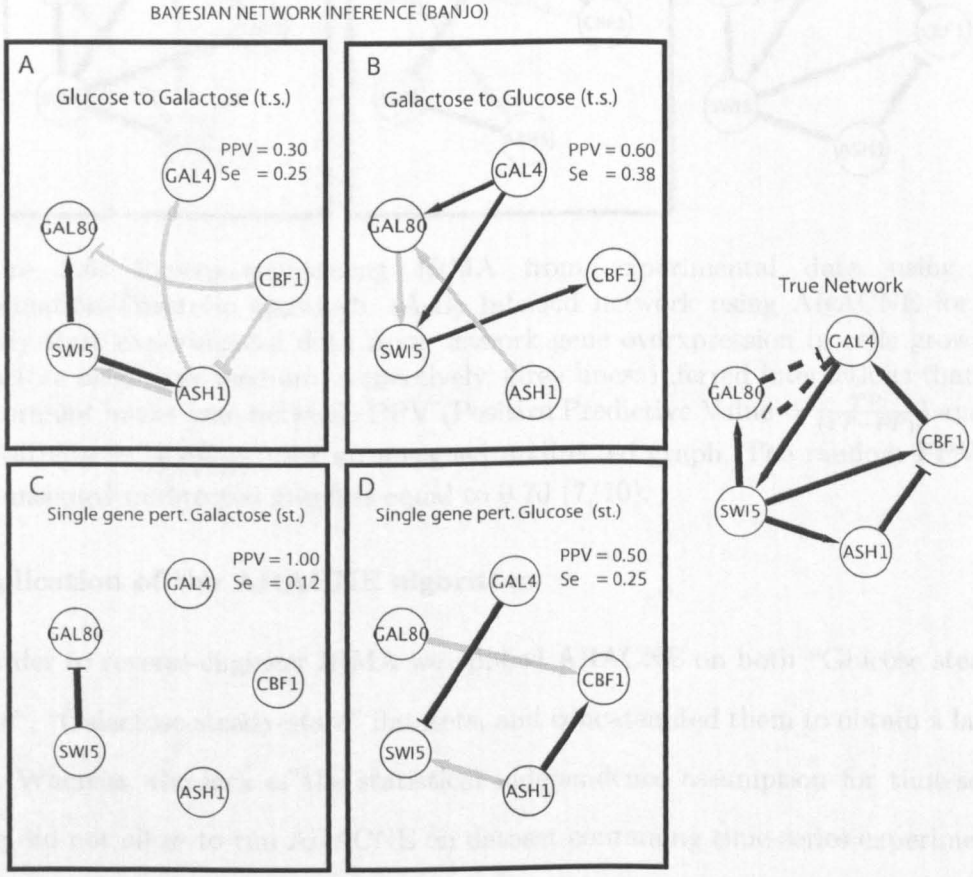


Figure 3.5: Reverse-engineering IRMA from experimental data using the Bayesian Network approach. (A-B) Inferred network using Banjo for the switch on and switch off time-series. Gray lines: inferred interactions that are not present in the true network. PPV (Positive Predictive Value = $\frac{TP}{(TP+FP)}$) and Se (Sensitivity = $\frac{TP}{(TP+FN)}$)) values for an unsigned directed graph. The random PPV is equal to 0.40. (C-D) Inferred network using the Banjo for the steady-state experimental data from network gene overxpression in cells grown in galactose or glucose medium, respectively.

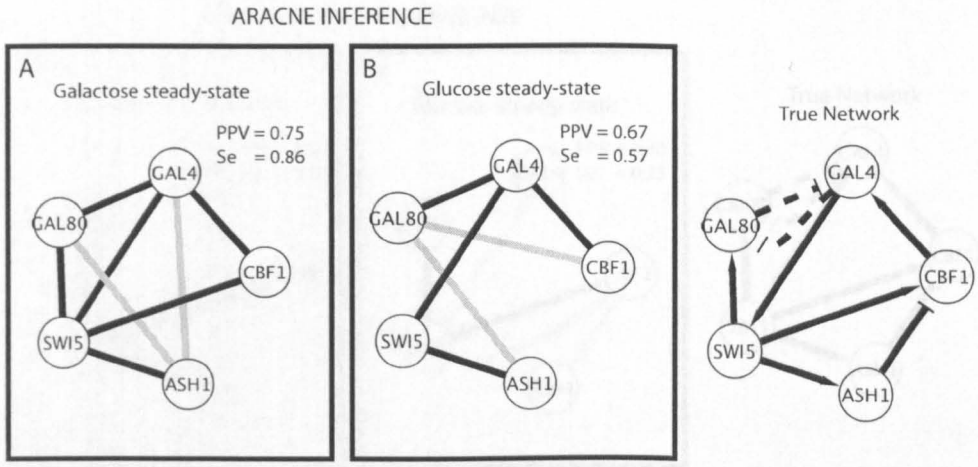


Figure 3.6: Reverse-engineering IRMA from experimental data using the Information-Theoretic approach. (A-B) Inferred network using ARACNE for the steady-state experimental data from network gene overxpression in cells grown in galactose or glucose medium, respectively. Grey lines: inferred interactions that are not present in the true network. PPV (Positive Predictive Value = $\frac{TP}{(TP+FP)}$) and Se (Sensitivity = $\frac{TP}{(TP+FN)}$) for an unsigned undirected graph. The random PPV for the unsigned undirected graph is equal to 0.70 (7/10).

Application of the ARACNE algorithm

In order to reverse-engineer IRMA we applied ARACNE on both "Glucose steady-state", "Galactose steady-state" datasets, and concatenated them to obtain a larger one. Whereas, the lack of the statistical independence assumption for time-series data did not allow to run ARACNE on dataset containing time-series experimental data. All the parameter were set to their default values. For instance, *Kernel width* and *Number of bins* are automatically detected by the software; no *threshold* and *p-value* among both MI values and MI P-value were used, respectively; *DPI tolerance* to remove false positive connections was left to its default value, *0.15*.

Application of the NIR algorithm

NIR [44] solves equation (3.4) to obtain the network matrix A from gene expression data. We considered a fixed number of regressors for each gene ($k = 2$), i.e. we assume that each gene can be regulated by a maximum of 2 other genes. The regressor set

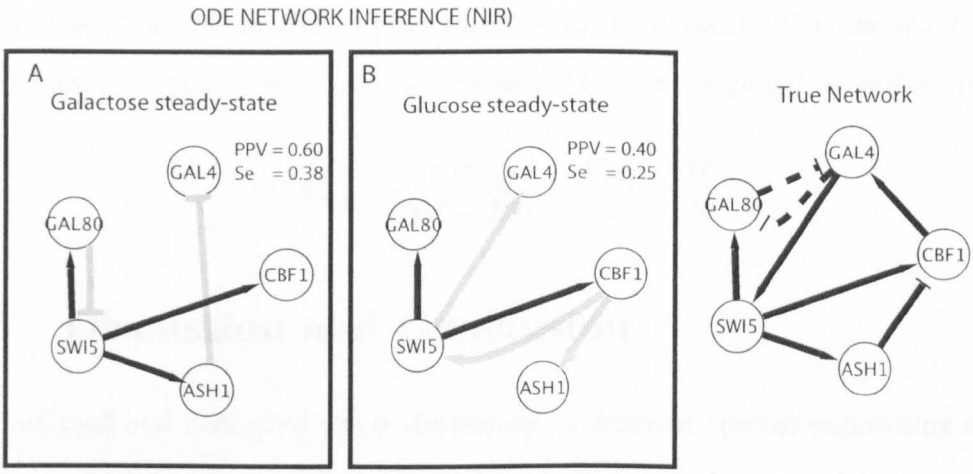


Figure 3.7: Panel A (B) report the inferred interactions by reverse-engineering IRMA network from steady-state data when Galactose (Glucose) is provided to the cells.

was chosen according the residual sum of square error (RSS) minimization criterion. The small size of the system (5 genes) allowed to exhaustively search for the best regressors. Perturbation values were set equal to 1. Results are reported in Figure 3.7.

Computation of the random performance

In order to compute the random PPV, that is the one of an algorithm that connects genes in the network randomly from a uniform distribution, we considered the expected value of an hypergeometrically distribuited random variable whose distribution function and expected value are, respectively:

$$P_x = \frac{\binom{M}{x}\binom{N-M}{n-x}}{\binom{N}{x}}, \quad E[x] = M\frac{\binom{N-1}{n-1}}{\binom{N}{n}} = M\frac{n}{N}.$$

In our case, N is the number of possible edges in the network; M is the number of true edges, n is the number of predicted edges. Then we computed as random PPV:

$$PPV_{rand} = \frac{TP_{rand}}{TP + FP} = \frac{E[x]}{n} = \frac{M}{N}.$$

3.5 Discussion and Conclusion

We analysed and compared the performances of different reverse-engineering algorithms on both in-silico and experimental expression data. In particular, concerning the simulated expression data, we used two classes of perturbations that we defined as "local" and "global". As outlined in the result section, for the local perturbation (Table 3.3) and the global perturbation (Table 3.4), there is a difference in the amount of information that is possible to recover following the two types of perturbations. Algorithms perform better when only a small subset of genes (possibly only one gene) is perturbed at a time.

Results on in-silico data are in line with respect to the results on experimental data. Since BANJO requires the estimation of a probability density function it can only be applied when many experiments are available. Moreover, BANJO is very precise but its sensitivity is very low. ARACNe performs generally better than BANJO, and even though it is not generally correct, it can also be applied to time-series dataset. ARACNe computes pair-wise "distances", and this allows it to run on larger dataset with respect to the other algorithms that relate a given gene to many others. NIR performs well also when the amount of data is small, it can be applied on steady-state data and performs well both on local and on global perturbations.

Chapter 4

Reverse-engineering gene networks from massive and heterogeneous gene expression profiles

4.1 Introduction

Tens of thousands of protein-protein, protein-DNA and protein-RNA interactions have been experimentally identified in mammalian organisms [60, 115]. However, they constitute only a small part of the complex network of regulatory interactions occurring in a cell. Due to the time required to experimentally identify specific interactions, efforts have been made to infer gene regulatory networks directly from gene expression profiles, using a variety of "reverse-engineering" algorithms [24, 9, 10, 12, 31]. Among the plethora of different approaches to reverse-engineering, the most successful, and generally applicable, are those based on information-theoretic approaches [9]. The network among genes is reconstructed by considering pairs of genes and checking whether the two genes in each pair are co-expressed across the experimental dataset. Co-expression can be measured either by correlation, or by a more robust measure, called mutual information (MI). As described in Chapter 2, a gene-gene "connection", thus inferred, is not necessarily a physical interaction between the protein (or RNA) products of the two genes, but can also imply a functional, but indirect, regulation.

Reverse-engineering becomes much more powerful as the number of gene expression profiles used to infer the network increases as shown in Chapter 3.5 [9, 75]. However, the requirement of using homogeneous gene expression profiles (i.e. from a specific cell type, tissue or condition) typically limits their number to the order of hundreds. Indeed, it is common belief that reverse-engineering a gene network by combining gene expression profiles from heterogeneous samples will generate too much noise, thus hampering the detection of biologically relevant signals. Our starting hypothesis was that, despite the extreme heterogeneity of gene expression profiles coming from different cell types, tissues, and conditions, it is indeed possible to infer a meaningful "consensus" gene regulatory network.

In this chapter we describe a new reverse-engineering algorithm able to use het-

erogeneous and massive sets of gene expression profiles to infer a gene network.

4.2 A new algorithm for reverse-engineering

4.2.1 Normalisation of Gene Expression Profiles

Gene expression profiles (GEPs) are usually organized in experiments. GEPs within experiments are normalized together using standard techniques described in 2.2.4. This procedure yields GEPs that are normalized within each experiment, but that are not normalized across experiments. This means that the expression level of a gene in two different experiments, although with the same absolute value, can refer to two different levels of expression. Normalised GEPs cannot therefore be merged across experiments. One way to obtain a single set of comparable GEPs from different experiment is to consider them as part of the same experiment, hence considering raw expression data for each GEP, and then normalise them all at once. Since standard normalization techniques are both difficult to apply to large datasets and could yield incorrect results due the assumptions underlying these techniques, we decided to proceed differently.

In order to derive numerically comparable measures of gene expression for different experiments, we discretized normalised GEPs within each experiment via quantile discretization [71]. This method is based on equal frequency binning. Here, the expression values of all arrays within an experiment are discretized into a predetermined number of bins (three). The three bins, with equal number of values, are determined using the three quantiles of the normalized expression values as cut points. Each expression value is then replaced by an integer value corresponding to the bin it falls into.

The choice of the number of bins may affect the subsequent results of the reverse-engineering analysis, however as shown in Section 5.3 this is not the case in our

settings.

4.2.2 Mutual Information

As described in Chapters 2 and 3, mutual information (MI) is a pseudo distance between probability distributions; it measures the amount of information two random variables share. We decided to use a MI reverse-engineering approach due to the good advantages of this approach over the others reverse-engineering approaches as detailed in Chapter 3.

Genes can be seen as random variables and their profiles as a random process. Once gene expression profiles are properly discretized into bins, it is possible to compute MI for each couple of genes by merging all the discretised data together. Specifically, for each pair of probes, we considered two discrete random variables $I \in \{1, ..., 3\}$ and $J \in \{1, ..., 3\}$ respectively describing the discretised state of the two genes. In this context MI can be defined as:

$$MI_{ij} = \sum_{i=1}^{3} \sum_{j=1}^{3} \pi_{ij} \log \frac{\pi_{ij}}{\pi_{i+}\pi_{+j}}, \tag{4.1}$$

where π_{ij} represents the joint probability $P(I = i, J = j)$ and $\pi_{i+} = \sum_j \pi_{ij}$ and $\pi_{+j} = \sum_i \pi_{ij}$ are respectively their marginal probabilities $P(I = i)$ and $P(J = j)$. Let n_{ij} be the counts of the outcomes of pair $(I = i, J = j)$ in the discretized sample set, then the frequency $\hat{\pi}_{ij} := \frac{n_{ij}}{n}$ gives rise to a probability matrix $\hat{\Pi} = (\hat{\pi}_{ij})$ of dimension 3x3, where n is the number of experiments where both I and J appear. These values can be used as an estimate of the unknown true joint probabilities matrix $\Pi = (\pi_{ij})$. This leads to a point estimate of MI equal to

$$\widehat{MI}_{ij} = \sum_{ij} \frac{n_{ij}}{n} \log \frac{n_{ij}n}{n_{i+}n_{+j}}. \tag{4.2}$$

Parallel implementation of the MI

The computational complexity of our algorithm is $o(N^2 \cdot K)$ where N is the number of genes (or probes in the case of microarrays) and K is the total number of expression profiles. Typically, both the number of genes measured and the number of available GEPs in public repositories are in the order of 10^4. Due to the high computational cost, we decided to implement a parallel version of the algorithm to reduce computational time. The parallel algorithm distributes the gene expression profiles among several computing processes. Each process gets N/p probes where p is the number of processes available and N is the total number of probes. The processors are named P_0 to P_{p-1} and logically organized in a topological ring where i follows j if $i > j$ and P_0 follows P_{p-1}.

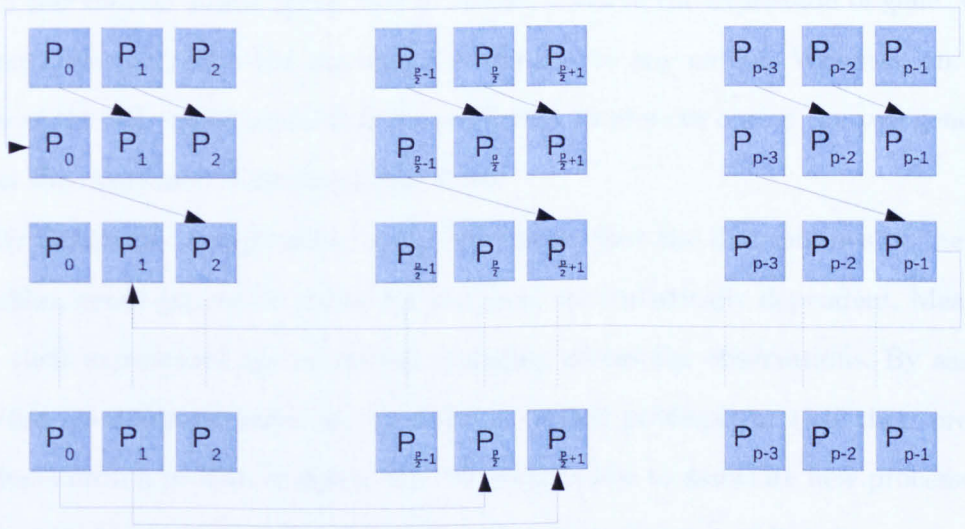


Figure 4.1: Processors are organized in a topological ring. Data is passed among processors according to their topological organization.

At the beginning of the computation each process calculates the discretization of its own gene expression profiles and computes the mutual information for each pair of its N/p probes. At the first communication step each process sends its probes to the following process and computes the MI for all the pairs of probes where the first is the local probe while the second is the received probe. At the i^{th} communication

step process P_j receives probes from process $P_{(j-i)mod(p)}$. The algorithm completes in $\lfloor \frac{p}{2} \rfloor$ communication steps. The parallel algorithm has been implemented in C using the MPI standard. The code is available in Appendix B. The algorithm has been executed on 105 processors of an HP XC6000 Cluster with Itanium 2 biprocessors nodes and a Quadrics ELAN 4 network.

4.2.3 Biological interpretation of the Mutual Information

In information theory the MI of two random variables, X and Y, measures the amount of information they share, that is the quantity on information (expresses in bits) we know about variable X by only observing Y. In this this work the MI is computed for each pair of genes from their expression profiles. A zero value of the MI in this context would mean that if we only observe the expression of gene X we are not able to predict the expression of gene Y to any extent. Whereas, an high value of the MI would mean that it is sufficient to observe one of the two genes to assess the expression behavior of the other.

By definition an high value of the MI means that the distributions of the two variables, genes expression values for instance, are statistically dependent. Meaning that their expressions are coherently changing across the observations. By assuming that co-expressed genes are involved in related processes or that their protein products form a protein complex, the MI would allow to associate new processes or interactors to genes or proteins for which this information is unknown.

4.3 Application to simulated data

4.3.1 Simulated Dataset

We applied our reverse engineering algorithm on simulated in-silico gene expression data described in Chapter 3 [9]. Specifically, the in-silico dataset we used consists

of 20 networks of 10, 100 and 1000 genes. For each network, we used a set of 100 expression profiles generated by perturbing one gene at the time (local perturbation) and a the set of 1000 gene expression profiles obtained via global perturbation, as reported in Section 3.2.1.

Moreover, in order to make this in-silico dataset comparable to real gene expression profiles coming from different experiments, and therefore with values not comparable across experiments, we proceeded as follows. For each gene network, we subdivided each set of the 1000 in-silico expression profiles into 10 different subsets of 100 expression profiles each. We then added a constant value to each of the subsets to simulate heterogeneity in the data. This dataset was mainly used to test the efficacy of the quantile discretization (see 4.2.1 for details). We refer to this simulated dataset as Dataset A.

A second set of simulated expression profiles (using a continuous differential equation model) were used to test and compare this reverse-engineering algorithm with other available algorithms. This set of simulated expression profiles is part of the fourth challenge of the third edition of the Dialogue for Reverse Engineering Assessment and Methods (DREAM) [91]. In this set, data are generated from 5 networks of genes (sub-networks of biologically validated gene networks, 2 for E. Coli and 3 for S. Cerevisiae). Regulation dynamics were assumed to follow a thermodynamic model. Gaussian noise was added to the generated expression profiles.

Each network consists of 100 genes. Two sets of 101 expression profiles were generated for each network. The j^{th} expression profile is simulated by deleting a copy of the j^{th} from the gene network (heterozygous mutant); the 101^{th} expression profile corresponds to the response of the wild type (original) network. In a second version of this dataset, expression profiles are simulated by removing both the copies of a gene (homozygote null mutant). We refer to these datasets as Dataset B and C.

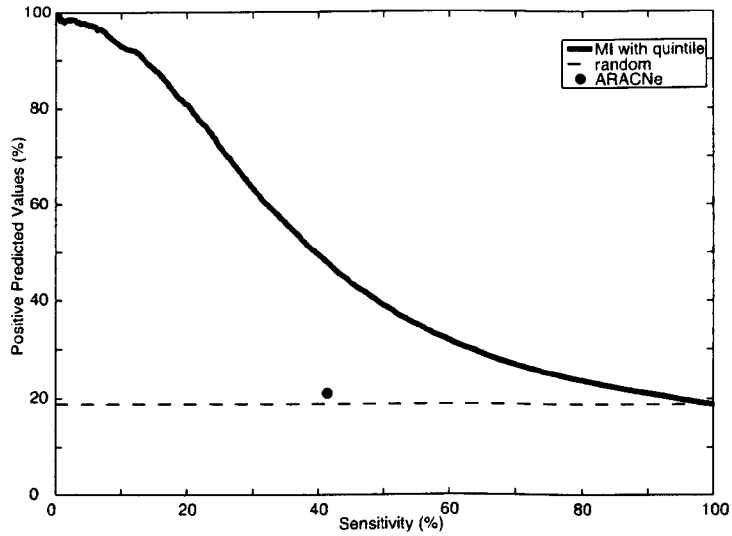


Figure 4.2: Gene network inference performances on Dataset A.

4.3.2 Performance on simulated data and comparison with state-of-the-art reverse-engineering algorithms

As described in Chapters 2 and 3, information theoretic approaches to reverse-engineering gene regulatory network have been first applied more than a decade ago [21]. The very first practical application came with ARACNe [77], successfully applied to infer the regulatory network in human B-Cells [12]. Our approach differs from ARACNe in the way of computing the MI: ARACNe uses the continuos version of the formula described in Section 2.2.2 and in Section 3.1.2, where the distribution function is obtained by fitting a gaussian distribution over the set of expression values; our method applies the discrete version of the Equation 4.1.

We applied ARACNe [77] on Dataset A. Since ARACNe needs a single normilised dataset to run properly, and since each network in Dataset A consists of 10 different expression subsets, we first applied ARACNe on each subset and then considered the union of the gene networks thus inferred in further analyses. On Dataset A, we also applied the new MI reverse-engineering approach described previously. We observed that the average precision of our approach across the 20 networks (True

Positives/ (True Positive plus False Positive)) is 47%, whereas ARACNe reaches a precison of 21%. For comparison, when ARACNE is run on all the 1000 in-silico expression profiles together, its precision is 66% [9]. This means that ARACNE is very good at inferring gene networks but only if gene expression profiles are coming from homogeneous samples, where standard normalization procedures can be applied.

Both the approaches were also applied on Datasets B and C. The set of algorithm predictions are compared by applying two scoring metrics: the Area Under the ROC curve (AUROC) that summarizes the tradeoff between the true positive prediction ratio and the false positive prediction ratio; whereas, the Area Under the Precision Recall curve (AUPR) that summarizes the Precision-Recall tradeoff. The edges predicted, sorted according to their weights, may only contain a subset of the true edges. The remaining (null predicted) true edges are then ranked randomly, and this procedure gives rise to a p-value associated to both the AUROC (P_{AUROC}) and the AUPR (P_{AUPR}) scores. The 5 p-values (one for each network), for each metric, are then evaluated together by taking their geometric means. The final *score* is computed as $-\frac{1}{2}\log_{10}(P_{AUROC} \times P_{AUPR})$ [91].

	Score	Overall P_{AUROC}	Overall P_{AUPR}	Pos
ARACNe	7.38	0.26	8.93×10^{-11}	10
our method	5.49	0.40	8.93×10^{-11}	11

Table 4.1: The performances of the two theoretical-information based algorithms were compared on a set of simulated expression profiles (Dataset B and C) [91]. We report the averaged performances obtained by analysing the heterozygous mutants and homozygote null mutant generated expression profiles. The "Pos" column contain the position of the two algorithms when compared with the performances of the other teams that participated at the DREAM3 (challenge 4).

Results are reported in Table 4.1. Given that the real synthetic network is directed and the inferred network has no directionality, and due to the symmetric nature of the mutual information, we forced the real network to be undirected before checking the algorithm performances.

It can be seen that even when expression profiles are optimal, that is data are not heterogeneous as in the case of simulated Datasets B and C, the two algorithm perform similarly and would rank among the top 10 in the DREAM competition.

4.4 Discussions and conclusions

Gene expression profiles are widely used to infer gene regulatory networks. The amount of biological data produced during the last decade was followed by a proliferation of reverse-engineering algorithm that make use of these data.

In Chapter 3, algorithms, under different assumption of network models, were tested and compared. Those algorithm can only be applied when the GEPs are comparable, that is, they are part of the same experiment, thus strongly limiting the number of expression profiles that can be analysed. Here, we presented a novel algorithm able to use expression profiles, even when they are not comparable.

Chapter 5

Reverse-engineering of human and mouse regulatory networks

5.1 Introduction

In this Chapter we describe the application of the new reverse-engineering approach presented in Chapter 4. The algorithm was run on two very large datasets of GEPs from human and mouse species. We show the reliability of the results by comparing the inferred gene networks with literature validated sets of inferred regulatory interactions. After the validation of the method we studied the properties concerning the topology of the recovered network and we show how these results are in line with previous findings. Moreover, we applied a community finding algorithm on the gene networks to discover groups of genes that are highly connected to each other and participate to the same biological function.

We show that the resulting networks can be used to discover protein-protein interactions, biological function and subcellular localisation of a protein, and to elucidate the function of a disease-gene.

5.2 Human and Mouse gene expression profiles

We collected 20,255 GEPs from 591 different experiments performed on a variety of human tissues, cell types, and conditions from the public microarray repository Array Express. A total of 22,283 different transcripts were measured, corresponding to the number of probes of the Affymetrix HG-U133A chip. From the same repository, we collected 8,895 GEPs from 614 experiment performed on a variety of mouse tissues, cell types, and conditions. Our aim was to exploit this massive datasets to yield consensus gene regulatory networks among the 22,283 human transcripts, and 45,101 mouse transcripts, using the reverse-engineering approach described in Chapter 4.

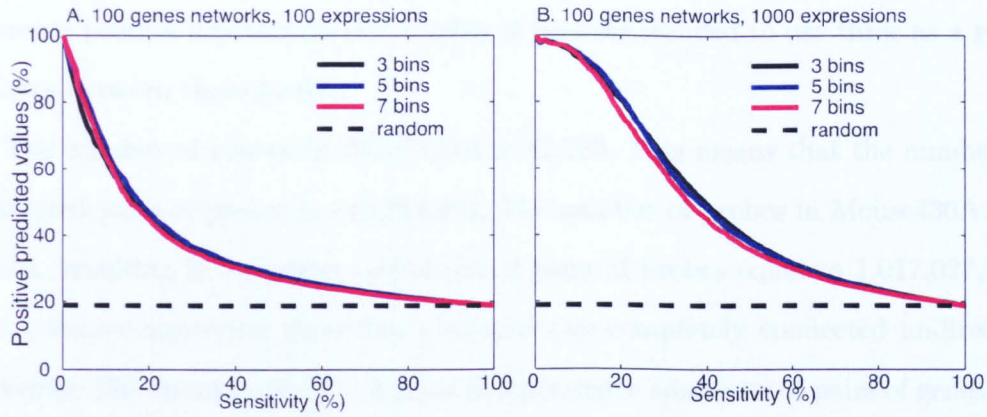


Figure 5.1: Comparison of the algorithm performance when changing the number of bins. Simulated networks of size 100 generated in [9] were used for the comparison. Data used in panel A are related to the inference from locally perturbed GEPs (100 data points). Whereas, in panel B were used simulated GEPs related to the global perturbation (1000 data points).

5.3 Application of the reverse-engineering algorithm

The algorithm presented in Section 4.2 was applied to the set of GEPs presented in the previous section. The algorithm was applied to the set of GEPs from human and mouse separately. The algorithm produced a MI value for each pair of probes in each of the two species. MI is always greater or equal to zero. MI equals zero only when the expressions of two probes are statistical independent. It is rare to find a pair of probes whose MI is exactly zero. Probes almost always share a signal, even though imperceptible, due to pre-processing and to finite sample size, which prevent computation of the exact MI value.

One of the algorithm parameters to be chosen is the number of discretisation states (or bins) the expression data have to be discretised into. It is possible to chose it arbitrarily according to data availability. However, the change in this parameter does not considerably affects the results as shown in Figure 5.1. Since the results are not changing significantly, and the computational complexity of the algorithm

inference process depends on the number of bins we decided to use three as a good balance between these factors.

The number of probes in HG-U133A is 22,283. This means that the number of indirected pairs of probes is 248,254,903. The number of probes in Mouse430A_2 is 45,101, resulting in a number of indirected pairs of probes equal to 1,017,027,550. The reverse-engineering algorithm produced two completely connected undirected networks. The assumption of biological independence among most pairs of genes imply the statistical independence of their expression profiles. Under this assumption, we adopted a null hypothesis that allowed us to select a threshold over the MIs. This threshold was used to discriminate statistically significant gene-gene connections. Under the null hypothesis of independence, the distribution of the MI follows approximately a Gamma distribution [46, 56].

We therefore fitted a Gamma distribution to the values of the MI across all the probes' pairs in human or mouse, using Maximum Likelihood estimation [49] are shown in Figures 5.2 and 5.6. We thus could assign a p-value to the MI of each gene pair and retained only those MI with a p-value < 0.01. This correspond to a threshold for MI equal to 0.04 for human and 0.025 for mouse.

We could have further pruned the networks by using one of the recently proposed schemes, such as the CLEAR method [40] and the Data Processing Inequality (DPI) method introduced by [77] and described in Section 3.1.2, but we decided against it, since we wanted to keep as many interactions as possible to have a broader overview of gene function and regulation. We were not interested in identifying 'direct' interactions as done in [12], but we focused on the identification of "co-expressed" genes, i.e. both direct and indirect interactions.

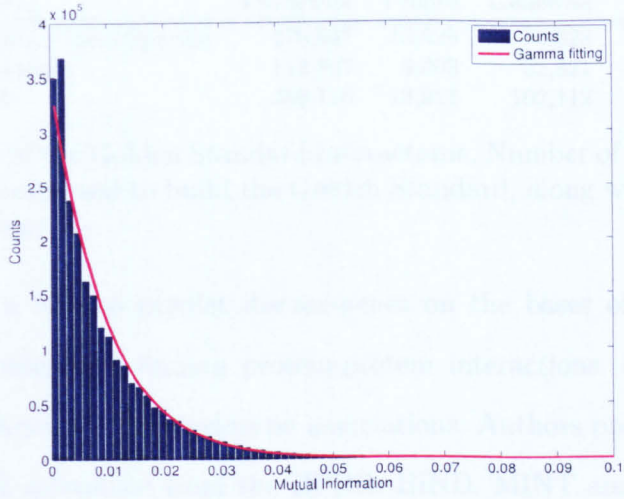


Figure 5.2: Fitting a Gamma distribution the histogram of the MIs recovered from human expression profiles along with the gamma pdf.

5.3.1 Human network

The number of significant connections is 4,817,629 connections (among 22,283 probes). To our knowledge, this is the largest and most comprehensive dataset ever used to reverse-engineer the human gene regulatory network.

To validate at least a subset of predictions, we generated a reference human interactome, Golden Standard (GoS), consisting of 105,588 experimentally verified interactions including protein-protein [115], transcription factor-target gene, and metabolic interactions [60] (Table 5.1).

In order to validate the biological relevance of the predicted interactions of our reverse-engineering approach, we built a golden standard (GoS) interaction network from the following interaction databases:

Reactome [60] is a curated knowledge-base of biological pathways; It represents a resource of core pathways and reactions in human biology. It collects a total of 32,821 metabolite-metabolite interactions[1] reported in literature.

[1] number of interactions obtained by mapping the specific dataset identifiers over the identifiers complained within the HG-U133A microarray model

ID	Source	Probewise	Probes	Genewise	UnigeneIDs
1	Cipher+TissueSpecific	270,937	13,525	80,828	7,766
2	Reactome	113,867	3,303	32,821	1,759
1&2	1&2	350,116	13,911	102,112	7,980

Table 5.1: Details of the Golden Standard Interactome. Number of interactions in the interaction databases used to build the Golden Standard, along with the cardinality of their intersection.

Cipher [115] is a tool to predict disease-genes on the bases of a computational framework that integrates human protein-protein interactions, disease phenotype similarities, and known gene-phonotype associations. Authors provide a protein interaction network assembled from the HPDR, BIND, MINT and OPHID protein-protein interaction databases. It accounts for 40,649 interactions[1].

Tissue Specific Protein Interactions [18] is a global human interaction network obtained by integrating data from 21 different sources to define a network of a total of 67,200 physical interactions[1].

Genes in the GoS network are identified with human UNIGENE IDs. On the other hand, microarray model HG-U133A identifies genes with probe IDs. In order to perform the comparison of the human interactome against the GoS network, we first mapped HG-U133A probes to 14,340 UNIGENE IDs using the annotation file provided by Affymetrix. Each gene-gene connection may correspond to multiple probe-wise connections because each gene can be associated to multiple probes. We therefore assigned to each gene-gene connection a MI value equal to the maximum MI among the corresponding probes-wise connections. The probe-oriented network was thus reduced to a gene-oriented one and made compatible with the GoS network.

Figure 5.3 shows the percentage of human predicted connections that were confirmed by the GoS interactome. Connections are ordered according to their MI, from the highest to the lowest. The network reaches a maximum of 90% of correct predictions, with an average precision of 32%. The percentage of correct connections, had these been randomly guessed, would have been equal to 0.0028%. The GoS interactome includes only a subset of the interactions occurring in a cell. Moreover, a high

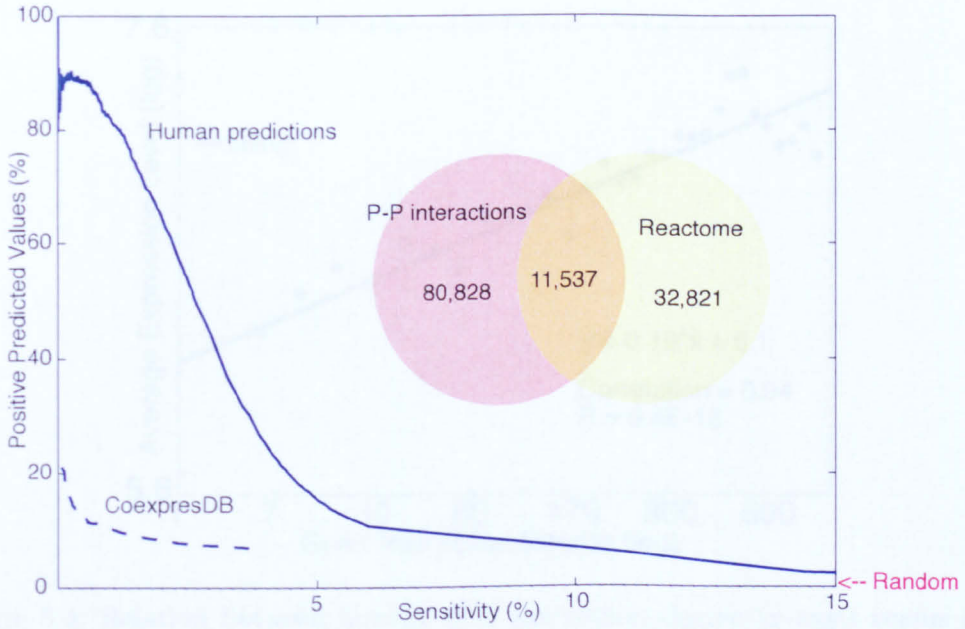


Figure 5.3: In-silico validation of the Human network compared with an experimental Golden Standard (GoS) interactome collected from different databases. The human network (blue-line) significantly outperforms correlation-based approaches (CoexpresDB dashed line). The Venn diagram reports the details of the interactions part of the GoS interactome.

MI does not necessarily imply physical interactions, as reported in the GoS, but more often functional relationships. Despite these limitations, the GoS interactome provides evidence of the biological significance of the inferred connections.

Figure 5.3 also shows the percentage of correctly predicted connections by CoexpresDB [84], a database of human co-expressed genes measured using a classic Pearson Correlation Coefficient (PCC). CoexpresDB performance is better than random, but much worse than our mutual information based approach.

The structure of the human network

The structure of the human network is typical of biological networks [82, 94]: it has an exponential degree distribution consisting of a large number of genes with very few connections, and very few genes with a large number of connections, termed hubs reported in Table 5.2.

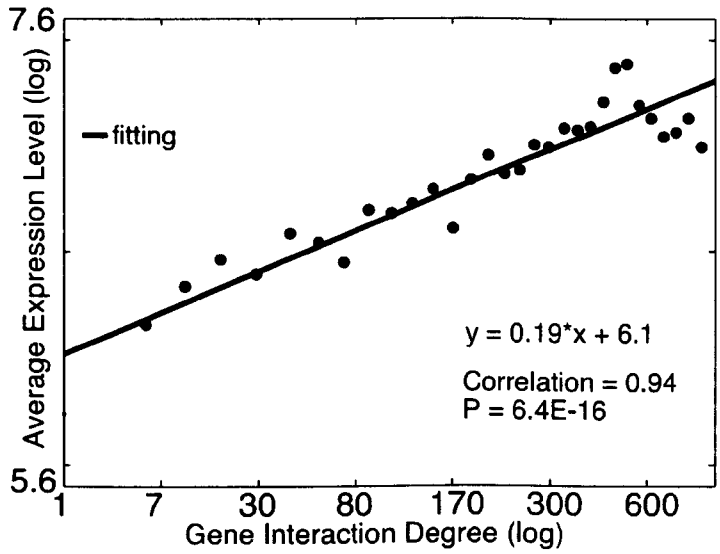


Figure 5.4: Relation between human gene connection degree (x-axis) versus gene expression level (y-axis). Genes were grouped in equally sampled (500 genes each) quantiles and the average gene connection degrees computed (x-axis). The log average expression level of the genes falling into each quantile is reported in the y-axis.

We observed that, as the number of connections of a gene increases, so does its average expression level as reported in Figure 5.4.

In order to relate gene degree to the average expression level of a gene, we proceeded as follow: since GEPs are normalized within experiments, and not comparable across experiments, we could not use the whole dataset to estimate the average expression of each gene. To this end, we used 618 human gene expression profiles from Array Express [88] measuring expression across a variety of normal tissues. GEPs were normalised together by applying the Affymetrix MAS5 algorithm. Average gene expression levels were then compared with gene connection degrees, i.e. the number of connections of a gene. Genes were first divided into equally sampled quantiles (500 each in human) then the average expression levels and the average gene connection degrees were computed. Due to the high degree variance in some of the quantiles, we decided to keep only those human quantiles where the connection

Probeset ID	Unigene ID	Gene Symbol	Connections
220531_at	Hs.710464	FLJ14126	5733
215366_at	Hs.585343	SNX13	5449
207969_x_at	Hs.169222	ACRV1	5364
216292_at	Hs.677419	—	5326
216900_s_at	Hs.10734	CHRNA4	5267
207140_at	Hs.37009	ALPI	5262
207685_at	Hs.533022	CRYBB3	5156
210923_at	Hs.104637	SLC1A7	5105
216739_at	Hs.589088	—	5010
1494_f_at	Hs.439056	CYP2A6	5002
214558_at	Hs.123034	GPR12	4989
215511_at	Hs.475018	TCF20	4979
215557_at	Hs.658129	—	4973
214923_at	Hs.712567	ATP6V1D	4957
220222_at	—	C8orf39	4919
32540_at	Hs.655661	PPP3CC	4918
211788_s_at	Hs.644635	TREX2	4870
206878_at	Hs.113227	DAO	4858
216440_at	Hs.658200	ERC1	4840
216159_s_at	Hs.654267	—	4798
208486_at	Hs.380681	DRD5	4778
34846_at	Hs.351887	CAMK2B	4720
215479_at	Hs.663736	—	4720
221466_at	Hs.673854	P2RY4	4715
207991_x_at	Hs.169222	ACRV1	4698
220671_at	Hs.639842	CCRN4L	4697
220826_at	Hs.677183	C21orf77	4670
202485_s_at	Hs.25674	MBD2	4653
216437_at	Hs.677301	—	4638
221199_at	Hs.302025	GFRA4	4636
207960_at	—	—	4592
215246_at	Hs.642978	LARP7	4590
210271_at	Hs.322431	NEUROD2	4578
216116_at	Hs.655006	NCKIPSD	4575
221378_at	Hs.248204	CER1	4570
206256_at	Hs.2246	CPN1	4566
211314_at	Hs.591169	CACNA1G	4563
221460_at	Hs.258574	OR2C1	4559
217293_at	—	—	4542
220082_at	Hs.192927	PPP1R14D	4529

Table 5.2: Human hub genes. Top 40 most connected human genes.

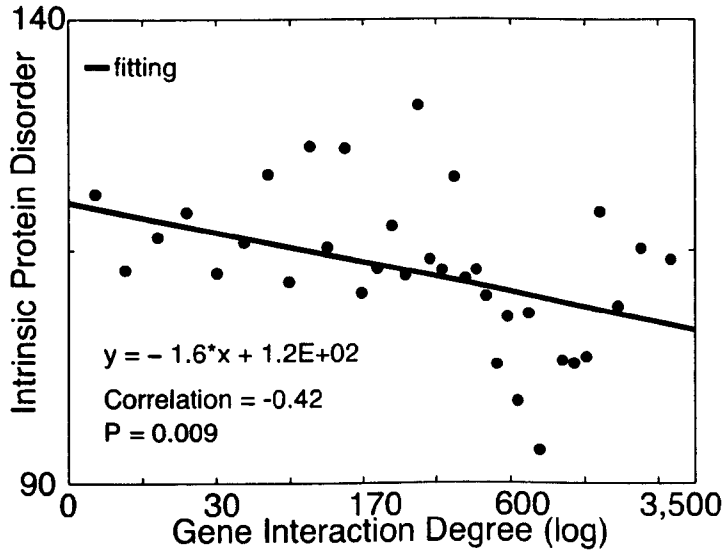


Figure 5.5: Relation between the number of connections associated to a gene (x-axis) and its dosage sensitivity (or Intrinsic Protein Disorder, y-axis) in human.

degree standard deviation was less 30.

On the contrary, the intrinsic protein disorder [70] of the protein product of a human gene significantly decreases ($P = 0.009$) as its connection degree increases, as shown in Figure 5.5. Protein disorder, defined as the length of the unstructured part of a protein, is an important determinant of gene dosage sensitivity [113].

In order to relate gene degree to the average protein disorder of a gene, we proceeded as before, but this time we used GlobPlot [70] to compute the protein disorder for each gene. We thus obtained a protein disorder score for 12,494 human genes (19,860 probes).

5.3.2 Mouse network

We repeated the same procedure described in Section 5.3.1 to infer the mouse network. The number of probes in Mouse430A_2 is 45,101. This means that the number of indirected pairs of probes are 1,017,027,550. However, the number of significant

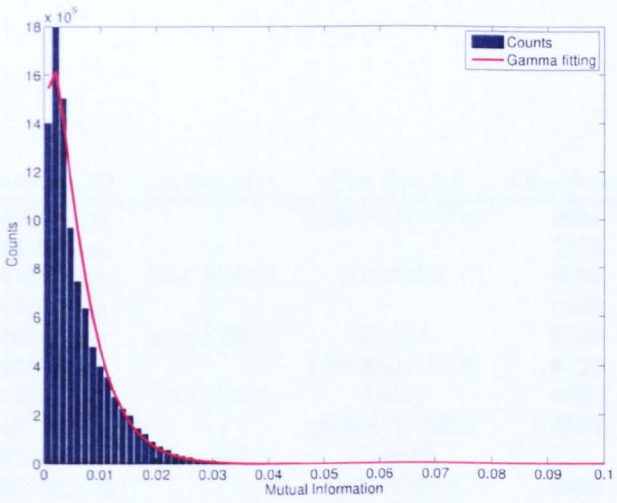


Figure 5.6: Fitting of a Gamma distribution over the MIs. The figure shows the histogram of the MIs recovered from mouse expression profiles along with the Gamma pdf.

connection are 14,461,095 (Fig. 5.6).

The structure of the mouse network is very similar to the human network: it has an exponential degree distribution consisting of a large number of genes with very few connections, and very few genes with a large number of connections (hubs) reported in Table 5.3.

Degree of connection and gene expression

Also for the mouse network, the number of connections of a gene increases, so does its average expression level as reported in Figure 5.7.

In order to relate gene degree to the average expression level of a gene, we proceeded as follows: since GEPs are normalized within experiments, and not comparable across experiments, we could not use the whole dataset to estimate the average expression of each gene. To this end, we used 247 mouse expression profiles from Array Express [88] measuring expression of genes across normal tissues. GEPs were normalized together by applying MAS5 algorithm. Genes average expression levels

Probeset ID	Unigene ID	Gene Symbol	Connections
1433070_at	—	5830433M15Rik	8031
1443066_at	—	—	7130
1443681_at	Mm.189222	AI595406	6822
1422426_at	—	—	6789
1460421_at	Mm.450554	Zfp133	6750
1433161_at	—	1700008N17Rik	6721
1437940_at	Mm.22879	Apba1	6653
1454428_at	—	4930467D19Rik	6582
1419802_at	Mm.249115	Ccdc12	
1445183_s_at	—	D7Ertd523e	6388
1454316_at	—	5830426C09Rik	6357
1431762_at	Mm.274255	Htra3	6329
1441838_at	Mm.280913	Ercc1	6276
1432949_at	—	5330421F21Rik	6253
1459088_at	—	C79557	6169
1453953_at	—	9130015A21Rik	6066
1425858_at	Mm.196580	Ube2m	5974
1430795_at	—	5830407F19Rik	5960
1433189_at	—	4933433N18Rik	5953
1453626_at	—	3930402G23Rik	5943
1437778_at	Mm.436700	Rbm15b	5930
1446335_at	—	—	5925
1442462_at	Mm.459149	—	5918
1450619_x_at	—	—	5913
1433245_at	—	6720475M21Rik	5876
1446650_at	—	—	5856
1429957_at	Mm.30967	Krtap26-1	5855
1445935_at	Mm.380510	—	5832
1425946_at	Mm.458189	Gstm7	5831
1453903_at	Mm.250432	4930503B20Rik	5813
1447246_at	Mm.122885	—	5772
1430919_at	Mm.444732	4930525F21Rik	5730
1439124_at	Mm.152120	Wdr91	5712
1431455_at	Mm.158563	Tmem30c	5702
1430429_at	Mm.28864	Pgs1	5696
1433221_at	—	2610311E24Rik	5680
1459696_at	Mm.216590	Fry	5635
1459206_at	—	—	5624
1421726_at	Mm.197568	Ap4b1	5594
1433400_at	—	5033405D04Rik	5592

Table 5.3: Mouse hub genes. Top 40 most connected mouse genes.

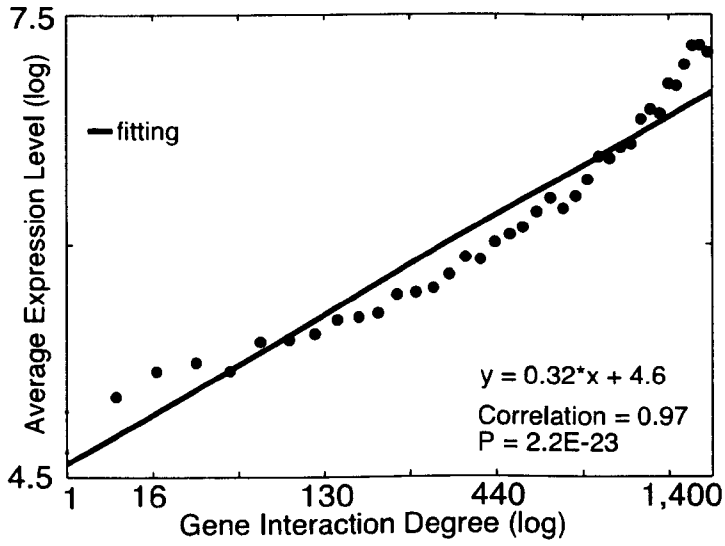


Figure 5.7: Relation between gene connection degree (x-axis) versus gene expression level (y-axis) in mouse. Genes were grouped in equally sampled (1000 genes each in mouse) quantiles and the average gene connection degrees computed (x-axis). The log average expression level of the genes falling into each quantile is reported in the y-axis.

were then compared with gene connection degrees, i.e. the number of connections of a gene. Genes were first divided into equally sampled quantiles (1000 each in mouse) then the average expression levels and the average gene connection degrees were computed. Due to the high degree variance in some of the quantiles, we decided to keep only those mouse quantiles where the connection degree standard deviation was less 50.

As for the human network, we observed that the intrinsic protein disorder [70] of the protein product of a mouse gene significantly decreases ($P = 0.009$) as its connection degree increases, as shown in Figure 5.8. Protein disorder, defined as the length of the unstructured part of a protein, is an important determinant of gene dosage sensitivity [113].

In order to relate gene degree to the average protein disorder of a gene, we used GlobPlot [70] to compute the protein disorder for each gene. We thus obtained a

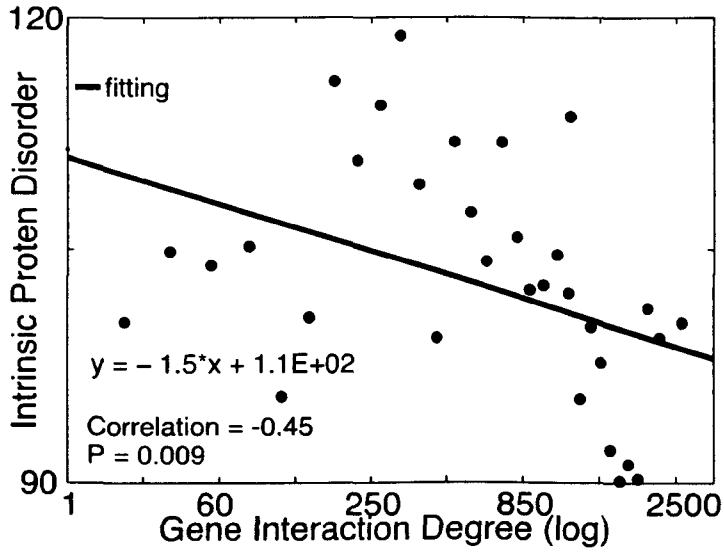


Figure 5.8: Relation between the number of connection associated to a gene (x-axis) and the its dosage sensitivity (or Intrinsic Protein Disorder, y-axis) in mouse

protein disorder score for 17,498 mouse genes (33,289 probes).

5.4 The modular structure of the human and mouse networks

A cell is able to regulate its complex behaviour thanks to groups of genes which perform different, but coordinated, functions to carry out a specific task. We asked whether we could find such functional modules within the inferred networks, which could reveal how the cell transcriptome is organized. We searched the network for modules, which are defined as "communities" and "rich-clubs" in network theory. A community is a group of genes highly inter-connected to each other, but with few connections to genes outside the group. A "rich-club" can be defined as a "community of communities", i.e. a group of closely inter-connected communities.

In order to identify communities, we represented the human network as a matrix, as shown in Figure 5.9A where the human adjacency matrix is reported. Each entry

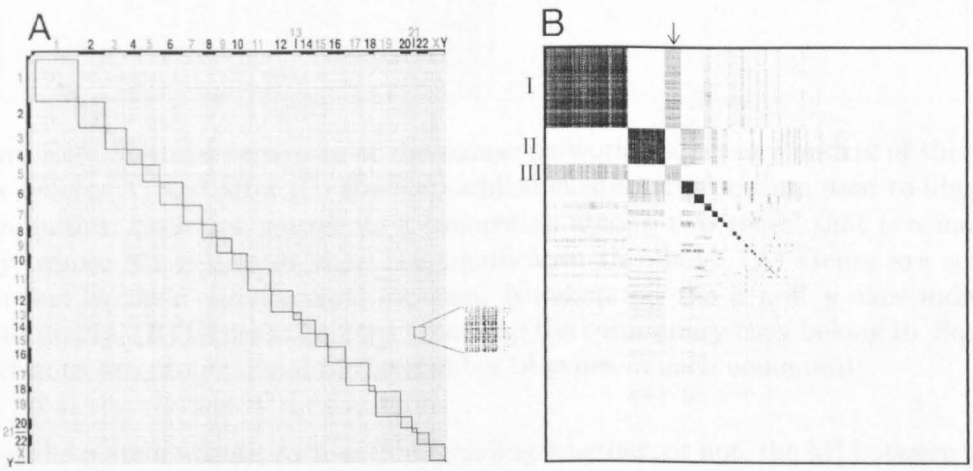


Figure 5.9: Modular structure of the human network. Adjacency matrix of the network before (A) and after (B) the hierarchical clustering procedure used to identify communities. Each dot represents a connection among two genes, that is a matrix entry, whose MI is greater than the significance threshold. (A) Genes are sorted according to their chromosomal location. Numbers on the x and y axes indicate chromosomes. (B) Genes are sorted according the community they belong to. Square dimensions are proportional to the number of genes in each community. The inset shows an enlargement of an area of the adjacency matrix where single dots are visible.

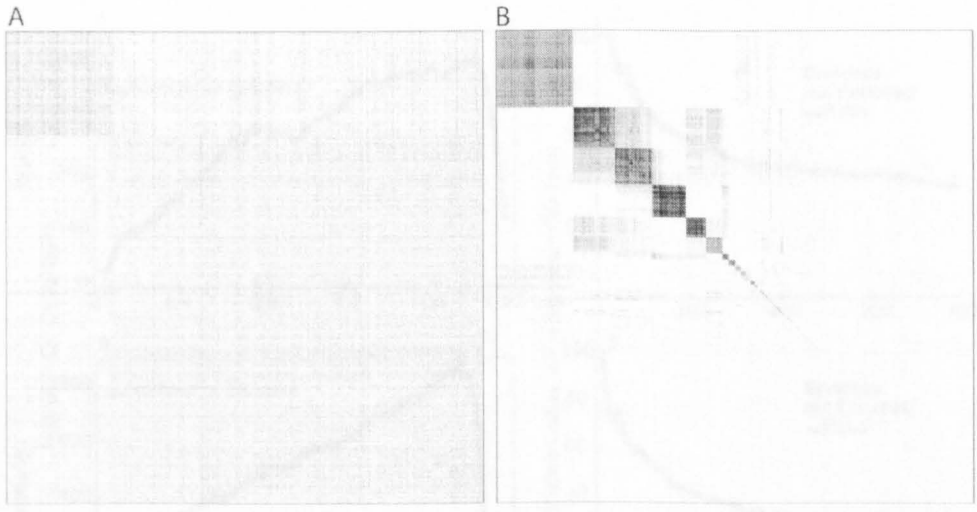


Figure 5.10: Modular structure of the mouse network. Adjacency matrix of the network before (A) and after (B) the hierarchical clustering procedure used to identify communities. Each dot represents a connection among two genes, that is a matrix entry, whose MI is greater than the significance threshold. (A) Genes are sorted according to their chromosomal location. Numbers on the x and y axes indicate chromosomes. (B) Genes are sorted according the community they belong to. Square dimensions are proportional to the number of genes in each community.

m_{ij} of the matrix was set to 1, or 0, depending whether, or not, the MI between gene i (in the i^{th} row) and gene j (in the j^{th} column) was significant. We observed that genes lying within the same chromosome (squares in Fig. 5.9A) tend to be connected to each other more often than what would be expected by chance.

In order to identify communities from the adjacency matrix, we applied a classic hierarchical clustering algorithm with the average linkage distance on the matrix describing the network using as a distance the "*Jaccard*" metric. This is defined as the ratio between the number of common interactions between two genes (i.e. two rows or two columns being the matrix simmetric) divided by the total number of interactions. The clustering procedure is outlined in Section 5.4.1. The dendrogram, produced by the average linkage algorithm, was cut in order to maximize the number of clusters with more than 4 nodes (Fig. 5.11A,B). At the end of this procedure, we identified 393 communities in the human network and 865 communities in the mouse network. Figure 5.9 and Figure 5.10 show the matrixes representing the human and

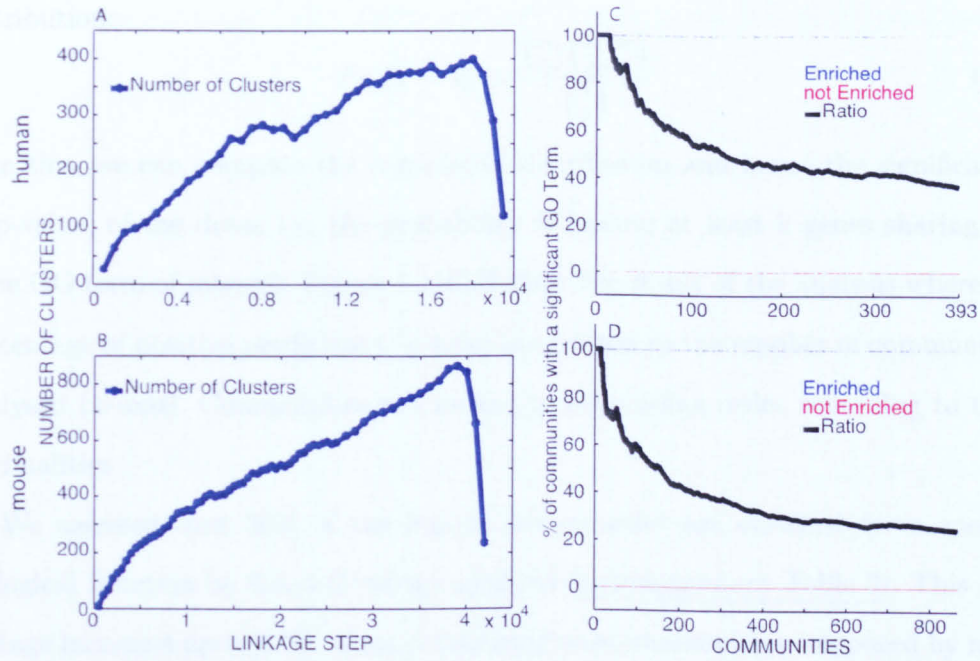


Figure 5.11: Communities validation through Gene Ontology analysis. Linkage step versus number of human (A) and mouse (B) communities with at least 4 nodes. Human (C) and Mouse (D) Communities were sorted according to their cardinalities. Each blue dot represents a community whose genes are enriched for a GO term whereas red dots express communities were no enrichment have been found.

mouse networks before (A) and after (B) the hierarchical clustering procedure (see Supplementary Table 2 for the list of communities). Communities appear as dark squares, with genes belonging to the same community grouped together, giving a striking check-board pattern.

We then checked whether communities were enriched for genes sharing a common biological function. To this end we applied Gene Ontology Enrichment Analysis on the list of genes in each community. GOEA is a commonly used technique that allows the identification of statistically over-represented Gene Ontology terms in a set of genes. Suppose to have a set of N genes, m of which are annotated as associated to a Gene Ontology term of interest. Suppose to draw a subset of n genes from the complete list of N genes, then the probability of obtaining k genes all sharing the same Gene Ontology term of interest follows an hypergeometric probability

distribution:

$$Pr(X = k) = \frac{\binom{m}{k}\binom{N-m}{n-k}}{\binom{N}{n}} \tag{5.1}$$

from this, we can compute the cumulative distribution and hence the significance, or p-value, of the draw; i.e. the probability of having at least k genes sharing the same GO term of interest. Figure 5.11C,D show the result of the analysis where the percentage of positive predictions (y-axis) are related to the number of communities analysed (x-axis). Communities are ranked in descending order according to their cardinalities.

We assessed that 36% of the human communities are enriched for a specific biological function by Gene Ontology analysis (Supplementary Table 3). This percentage increases up to 47%, when considering only communities composed by more than 10 genes. We also found that 6 out of 393 communities of the network are significantly enriched for disease-genes ($P < 0.05$, Gene Set Enrichment Analysis): community number 1, 11, 22, 40, 54 and 96. The most significant community, number 40, is composed by genes whose protein products localize to the *lysosome*, and is highly enriched for disease-genes involved in *lysosomal storage disorders*. Other examples include community 11, whose genes are related to *cell adhesion* and *extracellular matrix organization*, and include disease-genes causing developmental or cardiovascular defects; community 22, related to the *immune system* and including genes causing related disorders; community 54, composed by genes involved in *oxygen transport* and enriched for genes involved in hematological disorders. These "disease communities" could contain other yet unknown disease-related genes, and could be helpful in identifying candidate genes in disease-related loci.

We observed that communities interact with each other; for example community 1, enriched for *transmembrane receptor activity* ($P = 2.01 \times 10^{-35}$) interacts with community 3, enriched for *extracellular region* ($P = 7.33 \times 10^{-06}$, Fig. 5.9B, arrow), but not to community 2 involved in *RNA processing*.

5.4.1 Hierarchical clustering procedure to identify network communities

Hierarchical clustering methods [66] aim to group data items into a hierarchical set of clusters, organized in a tree structure, according to a specific distance function.

In order to apply one of these methods, it is necessary that all the distances between the items to be clustered are known, therefore a distance matrix must be calculated before the clustering procedure can be applied.

The hierarchical clustering algorithm starts with each item considered as a separate cluster. At each step, according to the distance matrix, the pair of closest clusters is merged into a new single cluster. The distance matrix is then updated by computing the distances between the new cluster and the others. This process iterates until all the items belong to a single cluster. It requires $N - 1$ steps where N is the number of the data items.

There are many possible distance functions that can be used in hierarchical clustering algorithms. We have adopted the unweighted average distance (UPMGA) [103]: the distance between any two clusters C_i and C_j is taken to be the average of all distances between pairs of items p in C_i and q in C_j:

$$\frac{1}{|C_i| \cdot |C_j|} \sum_{p \in C_i, q \in C_j} d(p, q).$$

To constructs the distance matrix starting from the network adjacency matrix we have used the Jaccard similarity coefficient. For two rows of the matrix, it is defined as the cardinality of their intersection divided by the cardinality of their union:

$$d(p, q) = \frac{|p \cap q|}{|p \cup q|}.$$

Parallel Implementation of the hierarchical clustering algorithm

The parallel algorithm distributes the gene adjacency matrix among the computing processes. Each process gets N/p rows of the adjacency matrix where p is the number of processes available and N is the total number of probes. Therefore each process stores a submatrix of the distance matrix with size $N/p \times N$.

As the distance matrix is symmetric, only the elements corresponding to the upper triangular half are calculated applying the Jaccard similarity coefficient to the rows of the adjacency matrix taken pairwise.

During the clustering process, the cluster index constantly changes: the singleton clusters are numbered from 1 to N, while each newly-formed cluster is assigned the index $N + I$, where I is the index of the iterative step ($1 \leq I \leq N - 1$).

At each iterative step I, each process finds the minimum of the distances locally. The global minimum distance of the closest clusters is found through an all-reduce operation and its indices are broadcasted to all processes. The computational time complexity is then reduced to $O({nClusters_I}^2/p)$, instead of $O({nClusters_I}^2)$, where $nClusters_I$, varying from N to 1, is the number of the clusters at step I. Therefore, the overall computational time complexity for this part of the algorithm is $O(\frac{N^3}{2p})$.

Each process updates the distances of the newly-formed cluster from the clusters that it stores. If the new cluster is obtained by merging the clusters i and j, then the new distances to be calculated are stored in the column and row i while the row and column j, from now on, will be excluded from the search for the minimum distance. Thus, each process just updates the portion of the column i (N/p elements) that it owns. The row i is instead updated through a *gatherv* operation in which all processes send its portion of the column i (N/p elements) to the root process that owns the row i (Figure 5.12). Therefore, the overall computational time complexity for this part of the algorithm is $O(\frac{N^2}{p})$. The parallel algorithm is summarized in Figure 5.13 and was implemented in C using the MPI standard..

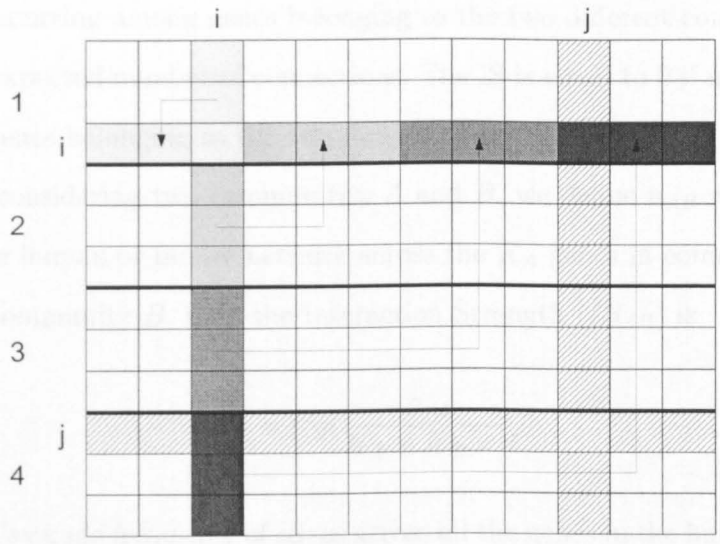


Figure 5.12: Distance matrix update example with $p = 4$ processes.

{compute distances of local clusters}
for $i = 1$ to $\frac{N}{p}$ **do**
  for $j = 1$ to $i - 1$ **do**
    $localD(i, j) \leftarrow d(i, j)$
  end for
end for
for $i = 1$ to $N - 1$ **do**
  {find local minimum distance}
  $(h, k) \leftarrow min(localD)$
  {all-reduce to find the global minimum}
  $(H, K, min\_rank) \leftarrow Allreduce(localD(h, k), MIN)$
  {broadcast of the indices of the global minimum}
  if $my\_rank = min\_rank$ **then**
    $Bcast(H, K)$
  end if
  {compute the distances of the new cluster}
  for $j = 1$ to $\frac{N}{p}$ **do**
    $localD(j, H) \leftarrow dist(C_H, C_K)$
  end for
  $localD(H, :) \leftarrow Gatherv(localD(0, H), \frac{N}{p})$
end for

Figure 5.13: Parallel hierarchical clustering algorithm

5.4.2 Communities of communities: "Rich-clubs"

In order to better elucidate community function and interactions among them, we defined the Interaction Strength (IS) between two communities as the number of

connections occurring among genes belonging to the two different communities, divided by the expected number of connections. The IS is equal to 0 if no connections exist among genes belonging to the two different communities.

Formally, considering two communities A and B, we define n_{AB} as the number of edges in the human or mouse network across the K_A genes in community A and K_B genes in community B, then the Interaction Strength (IS_{AB}) is

$$IS_{AB} = log(\frac{n_{AB}}{K_A \times K_B \times f}) \tag{5.2}$$

where f is the average frequency of edges across all the genes in the human or mouse network.

We then constructed a community-wise network by creating an adjacency matrix whose element in row i and column j is the IS_{ij} between community i and community j, if the $IS_{ij} > 0$.

We the used this community-wise adjacency matrix to identify rich-clubs, i.e. communities of communities. We computed the IS between all the pairs of 393 communities for a total of 77,028. Only 5,074 pairs of communities had an IS greater than 0. Similarly to the gene-wise network, also the community-wise network can be represented as a matrix. We can therefore apply a clustering procedure to group the communities into sets of highly interconnected communities ("rich-clubs") [42]. To this end, we applied a novel message-passing clustering algorithm [42] which is able to return the number of clusters without any user-speficied parameter, using as inter-community distances the ISs. We thus obtained 58 human and 227 mouse clusters of communities, i.e. rich-clubs (as defined in network theory). The process was iterated by considering the ISs among the new formed clusters (rich-clubs). Figure 5.14 report the human network of rich-clubs along with four magnifications. Figure 5.15 For a complete list of genes within each community refer to Supplementary Table 2.

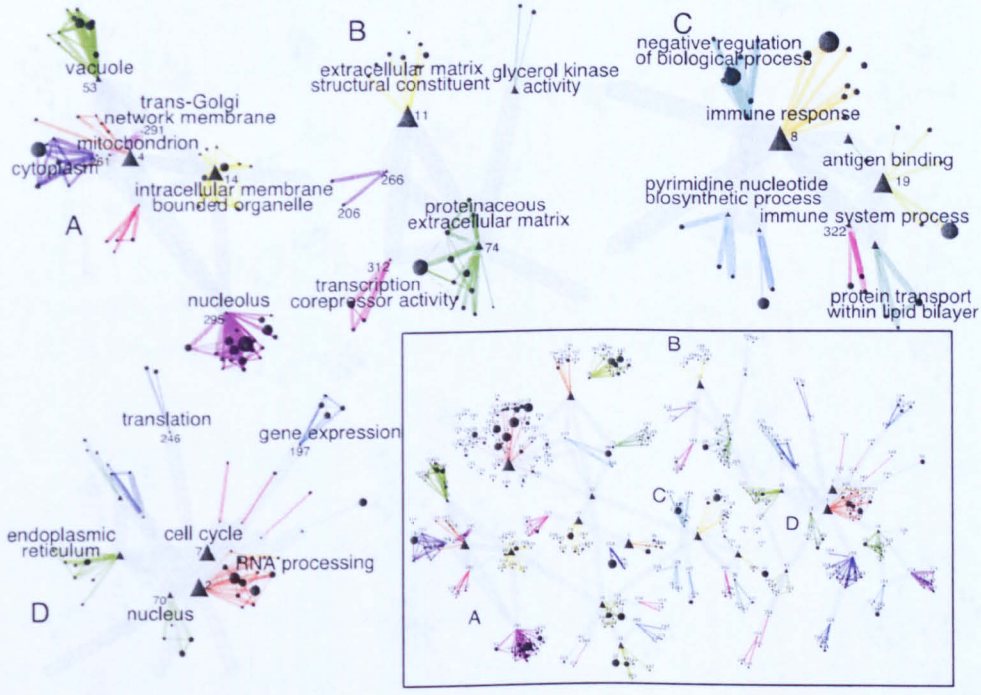


Figure 5.14: Community-wise human network. Each node is a community. A color and a number identify each rich-club (i.e. a group of highly interconnected communities). The width of each edge reflects the IS between communities. "Exemplars" are indicated by triangles. Examples of rich-clubs: (A) communities of genes involved in house-keeping functions: *gene expression* (rich-club 197), *translation* (rich-club 246), *RNA processing* (rich-club 2) and *cell cycle* (rich-club 7); (B) communities involved in the *extracellular matrix maintenance*, and *cell mobility*; (C) communities involved in *immune response*; (D) communities involved in *intracellular trafficking*.

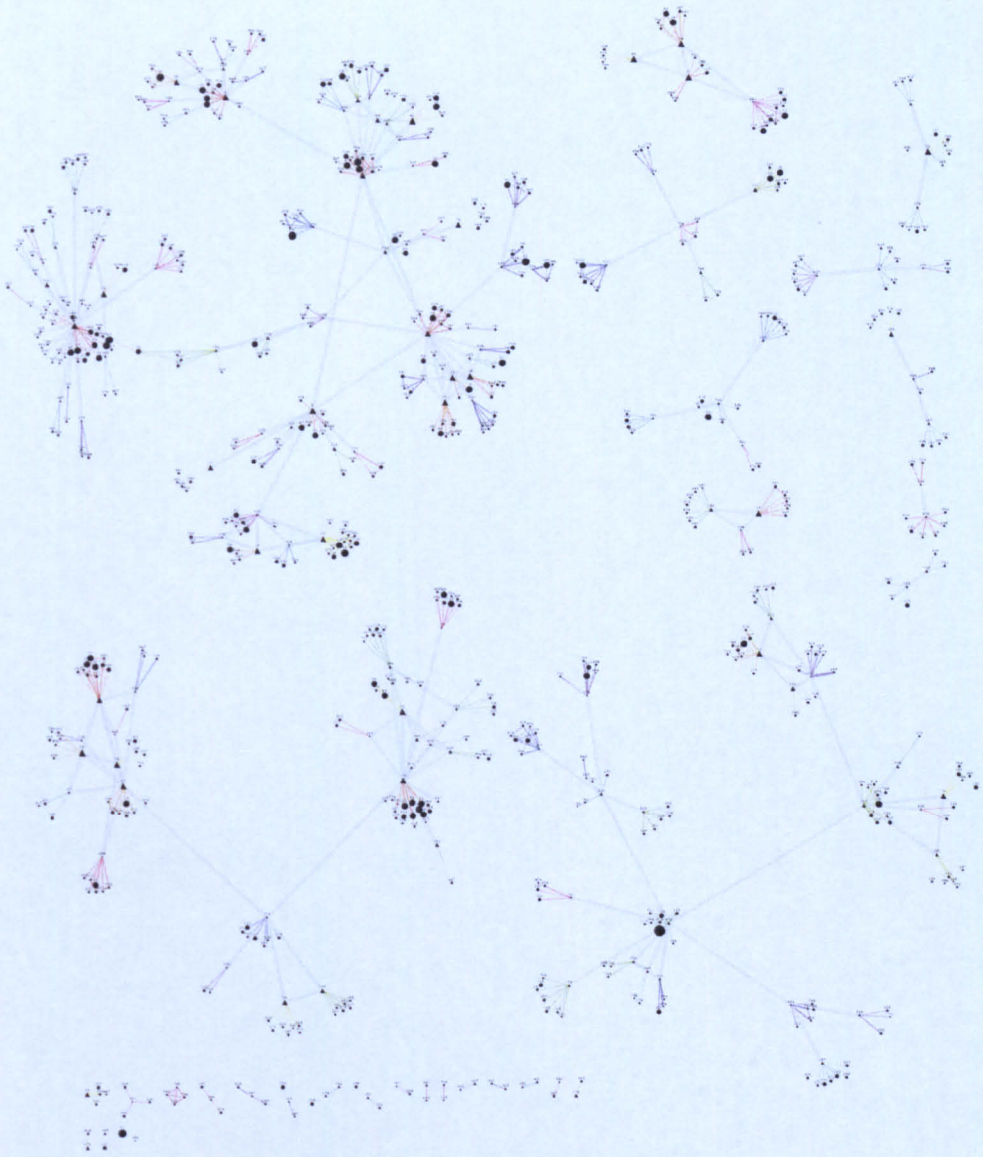


Figure 5.15: Community-wise mouse network. Each node is a community. A color and a number identify each rich-club (i.e. a group of highly interconnected communities). The width of each edge reflects the IS between communities and rich-clubs. "Exemplars" are indicated by triangles. The node of the network are identified with an integer number (see Supplementary Table 2).

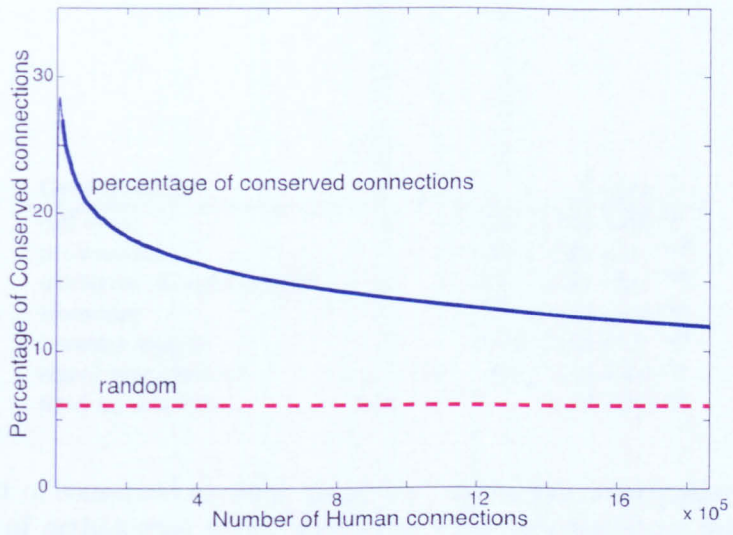


Figure 5.16: Percentage of mouse-conserved human connections. Connections are sorted according to their MI value. The peak of the curve is at 28%, and the genes involved in those conserved interactions are highly enriched for *cell cycle* $P = 1.0 \times 10^{-15}$

5.5 Connections tend to be conserved across species

To understand whether, and to what extent, connections among genes are conserved across human and mouse species, we first removed from the human network those genes without an ortholog in mouse, resulting in a "reduced" network of 11,318 genes. We then found that 218,700 connections (12%) were conserved in mouse as shown in Figure 5.16.

This percentage is in line with previous studies. In yeast, it has been reported that between 10% [95] and 30% [86] of protein-protein interactions occurring during the cell cycle of *Saccharomyces pombe* (fission yeast) and *Saccharomyces cerevisiae* (budding yeast) are conserved; another cross-species (fly and yeast) protein interaction study [8] resulted in a ratio of conservation ranging from 6% to 15%; in [93], the authors report a database of protein-protein interactions occurring among transcription factors in human and mouse, where the percentage of effective conserved interactions is about 16% (the estimated range is 34%-64% when taking into account

Gene ontology	human community	mouse community	orthologous genes	p-value
cell cycle	2	7	142	1.47×10^{-21}
proteasome	4	2	483	8.96×10^{-216}
oxidative phosphorylation	4	2	483	8.96×10^{-216}
ribosome	9	2	113	1.79×10^{-43}
immune system	8	8	114	2.02×10^{-129}
signal transduction	19	33	33	2.85×10^{-54}
lipid metabolism	120	51	7	1.61×10^{-13}

Table 5.4: List of conserved modules along with the human and mouse communities IDs, number of orthologous genes and the p-value associated to the Enrichment analysis.

the False Positive Rate of the experimental technique). A recent genome-wide analysis [4] integrating heterogeneous sets of experimental data (including 338 expression profiles in human and 1048 in mouse) showed a conservation of 15% of interactions between the two mammalian species.

We investigated whether communities found in the human network were conserved as communities in the mouse network. A human community was deemed to be conserved in mouse, if there was at least one mouse community composed by a significant fraction of genes orthologous to genes in the human community. Ninety-two out of 389 human communities (24%) were enriched in one, or more, mouse communities (Supplementary Table 5). We only considered those human communities with at least one mouse orthologous gene. It appears that the conservation at the community level (24%) is higher than conservation at the gene-gene level (12%) suggesting that many circuits embedded within the networks were conserved during evolution. Our results are in agreement with previous studies on the conservation of protein complexes across species [99]. Example of conserved communities are listed in Table 5.4: as expected many of the basic cellular processes are conserved [106].

We observed that human *community 4* and the conserved mouse *community 2*

are significantly enriched for genes involved in *proteasome* (human $p = 7.25 \times 10^{-22}$, mouse $p = 7.83 \times 10^{-08}$) but also for genes involved in *oxidative phosphorylation* (human $p = 1.36 \times 10^{-55}$, mouse $p = 9.16 \times 10^{-12}$). Therefore, genes belonging to these different pathways are co-regulated, and this co-regulation is conserved across species. Proteasome-dependent protein degradation represents one of the most expensive processes in term of energy; oxidative phosphorylation is a metabolic pathway that uses energy released by the oxidation of nutrients to produce ATP. Therefore it would make sense for these two pathways to be coupled.

5.6 Discussions and conclusions

In this chapter we have shown that massive and heterogeneous gene expression datasets can indeed be exploited to yield biologically relevant information on gene regulation at transcriptional level. We have identified groups of genes that tend to be highly "co-expressed" (communities) and that share a common biological function.

We observed that genes that are connected, i.e. co-expressed, in human tend to be also co-expressed in mouse, more than what is expected by chance; however this is not the general trend, since only 12% of connections are conserved. Albeit this estimate may be lower than the real one, due to false negatives and orthologous gene mis-assignments, our results are confirmed by other large-scale experimental studies in human and mouse [93, 4], and this percentage would drastically increase if we would consider both that only a subset of genes are orthologous, and that the threshold we selected imposes an upper bound. Our observation adds weight to the hypothesis that regulation of gene expression may be very different between species, even if they share a similar proteome.

The gene network structure is typical of complex network with the presence of hub genes with a large number of connections. Our finding suggests that hub genes are highly expressed and may have been selected to be less dosage sensitive, i.e. not

pathological when their expression is increased, as confirmed by their tendency to have a lower protein disorder.

We identified biologically relevant functional modules within the network, thus providing a modular view of the wiring diagram of a cell, going from genes to communities, and from communities to rich-clubs.

Chapter 6

Transcriptome organisation and gene function

The final goal of an inference process applied to biological dataset is to gain new insights on the biological mechanisms of the cells. These insights include identification of physical interactions among proteins (P-P interactions), prediction of the biological function of a genes, and so on. In this Chapter we experimentally validated a set of physical interactions predicted by connections in the human network, and we propose a new approach called "guilty-by-association", able to predict the biological processes of genes, or its cellular localization of the protein products. Moreover, we investigated the relationship between three-dimensional organisation of the chromatin in the nucleus and gene expression.

6.1 Biological validation of Protein-Protein interactions

Genome-wide inference validation was a difficult process to accomplish due to the lack of a golden standard network of interactions among genes. As outlined in 5.3.1

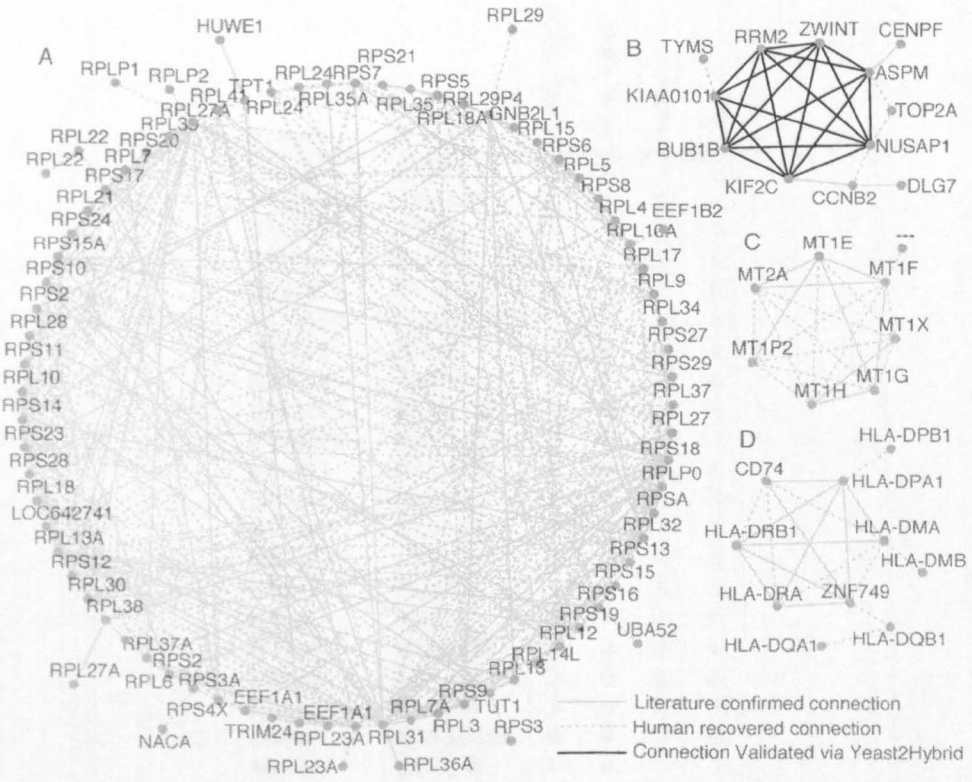


Figure 6.1: Subnetworks obtained by collecting the top 1000 connections with the highest MI within the network. Subnetwork (A) contains genes that codify for the *ribosomal protein complex*. Subnetwork (B) is enriched for genes involved in the *spindle checkpoint*, for clarity only a subset of interactions are shown. Subnetworks (C) is enriched for *metallothionein* genes, a family of low molecular weight, heavy metal binding proteins. Interestingly these genes are all present as a cluster on chr 16q13. Subnetwork (D) contains *major histocompatibility complex* proteins.

we built a golden standard (GoS) interaction network by merging different sources of interaction databases. Results are reported in Figure 5.3 and show that the inference performance reaches a maximum of 90% of correct predictions. The ROC curve highlight how the higher the MIs the more we can trust our predictions.

We investigated the identity of the top one thousand connections with the highest MI in the network (Fig. 6.1A and Supplementary Table 4). Forty percent of these connections, involving a total of 302 genes, were confirmed by the GoS interactome. An additional 13% of the connections were predicted among genes in the same gene-family, which, therefore, may well be functionally related, although not physically

Protein	Protein	strong signal	weak signal	negative	total
NUSAP1	RRM2	1	2	3	6
NUSAP1	ZWINT	1	3	2	6
NUSAP1	ASPM	3	3	0	6
NUSAP1	KIF2C	2	3	1	6
NUSAP1	BUB1B	2	4	0	6
NUSAP1	KIAA0101	0	3	0	3
ZWINT	RRM2	2	1	3	6
ZWINT	ASPM	3	0	3	6
ZWINT	**BUB1B**	2	0	4	6
ZWINT	**KIF2C**	1	0	2	3
ZWINT	KIAA0101	1	0	5	6
RRM2	KIF2C	1	1	4	6
RRM2	KIAA0101	1	0	2	3
RRM2	ASPM	2	0	4	6
RRM2	BUB1B	1	2	0	3
KIF2C	ASPM	2	2	2	6
KIF2C	**BUB1B**	1	2	3	6
KIF2C	KIAA0101	2	0	4	6
KIAA0101	BUB1B	1	0	2	3
ASPM	KIAA0101	0	0	3	3
ASPM	BUB1B	1	0	2	3
negative control		7	2	21	30
positive control		15	0	0	15
positive weak control		6	9	0	15

Table 6.1: Biological validation of Protein-Protein interactions. Protein-protein interactions that have been tested via Yeast two Hybrid assays. In bold we report the pairs of proteins that were known to interact from literature.

interacting.

In order to test the predictive ability of the network, we focused on the subnetwork (B) in Figure 6.1 consisting of 12 genes most of which (*CENPF*, *NUSAP1*, *KIF2C*, *BUB1B*, *ASPM*, *ZWINT*, and *CCNB2*) involved in *mitotic spindle checkpoint*, *chromosome motility*, and *mitotic progression* [59, 74, 92, 109].

According to the GoS interactome, three protein-protein interactions were known to occur among the genes in subnetwork (B), therefore we decided to verify whether the predicted connections, could be yet undiscovered protein-protein interactions. We selected only the subset of 7 genes (*NUSAP1*, *KIF2C*, *BUB1B*, *ASPM*, *ZWINT*,

KIAA0101, and *RRM2*), which according to our network formed a tight cluster of genes all connected to each other. We performed a series of yeast two hybrid (Y2H) assays to test a total of 21 connections (i.e. all the possible interactions among 7 proteins). According to the Y2H assay, 20 of these were positive. Since Y2H are known to be prone to false positive detections, we also experimentally estimated the precision (true positives over true positives plus false positives) of the Y2H assay by using appropriate positive and negative controls (SI). The estimated precision resulted to be equal to 77%, hence, at least 15 (= 77% of 20) of the experimentally identified interactions should be true positive predictions. This means that we can predict new p-p interactions with a precision of 75% (15 out of 21, Fig. 6.1, see Table 6.1 for the results of all the assays).

6.1.1 Yeast-two-Hybrid assays

The Yeast two Hybrid (Y2H) kit "ProQuest Two-Hybrid System" (Invitrogen) included the S. cerevisiae MAV 203 strain (MATα, leu2-3,112, trp1-901, his3Δ200, ade2-101, gal4Δ, gal80Δ, SPAL10::URA3, GAL1::lacZ, HIS3UAS GAL1::HIS3@LYS2, can1R, cyh2R), the bait vector pDEST32 and the prey vector pDEST22. The "Ultimate ORF" (Invitrogen) of the genes of interest were used to generate prey and bait plasmids using the GateWay technology and protein-protein interaction assays were performed according to manufacturer instructions.

6.2 Chromatin structure and gene expression

The three-dimensional structure of chromosomes in the nucleus brings into close spatial proximity loci that are far apart in terms of genomic distance [69]. However, little is known about the relationship between chromatin spatial structure and gene expression.

We investigated whether genes that are connected to each other according to

our network, were also physically close to each other at the chromatin level, in the cell nucleus. We used a recent comprehensive mapping of chromosomal loci physical interactions using an innovative "Hi-C" chromatin capture technology [69].

The physical interaction map was measured in 2 human cell lines (GM06690 and K562) [69]. Authors provide an intra-chromosomic [69] contact probability matrix at 100 Kilobases resolution, and a genome wide contact probability matrix at 1 Megabases resolution.

This map of physical interactions can be represented as a matrix (M_p), where each entry m_{ij} reports the probability of the i^{th} Mb of the genome to be in physical contact with the j^{th} Mb, according to "Hi-C" experimental results. Figure 6.4B is a graphical representation of this map for chromosome 19. In [69] authors observe that genes in the same chromosome that are close each to other in terms of genomic distance have an high probability to be physically in contact. By using the information of the connections among the genes that is stored in the inferred human network, we computed a "connection tendency" and compared it with the linear genomic distance.

Figure 6.2 shows the result of the analysis. Plots are associated to chromosome and are color coded according with the length of the chromosomes. Specifically, for each normilised chromosomal distance (y-axis), the number of connections among the genes that are within that distance is computed. This value is then divided by the expected number of connections (given the network) and the $log2$ is computed and reported in the x-axis. This quantity is referred to as the "connection tendency". The analysis shows that genes that are close each to other in terms of linear genomic distance, have an higher tendency to being connected in the network with respect to genes that are far apart. The analysis reported in Figure 6.2 is similar to the one performed in [69] where in place of connection tendency among genes they consider physical contact probability.

Communities in our network can be considered as functional modules consisting

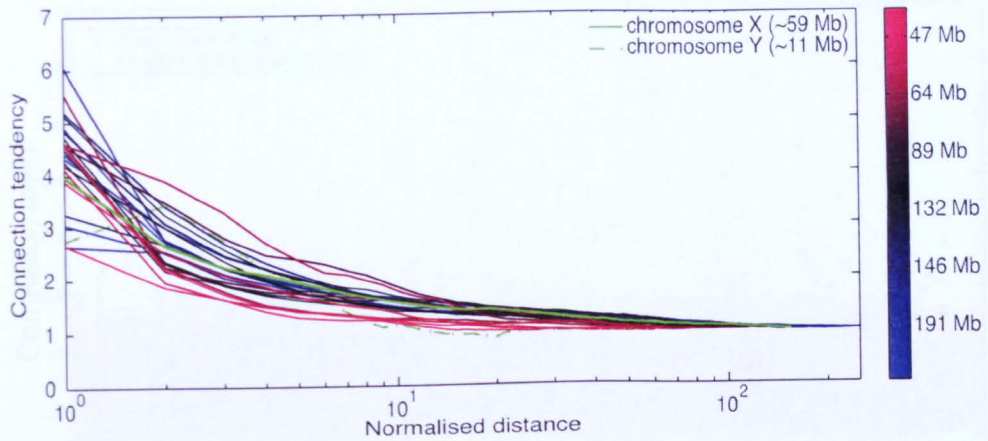


Figure 6.2: Connection tendency and genomic distance. "Tendency" (y-axis) of the human genes within a certain genomic distance to be connected according to our network. We computed a connection tendency as the number of connections among genes below a given genomic distance, divided by the expected number of connections for the same number of genes, independently of their distance. We color coded each chromosome, from red (shorter chromosomes) to blue (longer chromosomes). We see that the closer two genes are in terms of genomic distance, (x-axis, log-scale), the higher their tendency to be connected.

of genes whose expression is coordinated, and that carry out specific biological functions as observed in Section 5.4. We asked whether these gene modules could have a 'physical counterpart' in the cell. We therefore further investigated the correspondences between physical contact probability and connection tendency.

In [69], authors defined a correlation matrix C_p in which element (i, j) is the Pearson correlation between the i^{th} row and j^{th} column of M_p. This C_p matrix exhibits a strong 'plaid-pattern' as shown in Figure 6.4B. C_p is the correlation matrix that illustrates the correlation (range from 1 (blue) to +1 (red) in Fig. 6.4) between the intrachromosomal interaction profiles of every pair of 1 Mb loci along a chromosome (chromosome 19 in Fig. 6.4). The plaid pattern indicates the presence of two compartments within the chromosome. The computation of the correlation between each pair of chromosome loci is justified by the fact that if two loci are nearby in space, they reasonably will share neighbours and have correlated interaction profiles.

The analysis reported in Figure 6.3 was performed computing the contact corre-

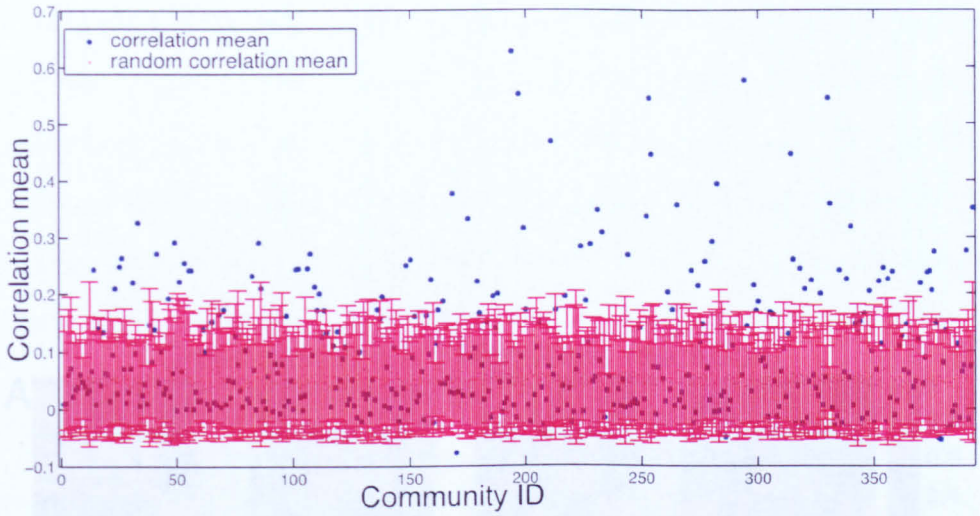


Figure 6.3: Intra-community physical contact (y-axis) expressed in terms of average contact correlation is plotted for each of the 393 communities (x-axis) identified in Section 5.4, and compared to the expected random correlation.

lation mean (from matrix C_p) among the chromosomal regions associated to each of the community found with the analysis presented in Section 5.4. To associate chromosomal regions to communities, we simple take all the chromosomal regions that contain at least a gene of the community. Then, for each community we averaged the correlations among the associated chromosomal regions (blue spots in Fig. 6.3). For each community we also checked for the random correlation by shuffling the labels of the genes in the community, while keeping the size of the community (red spots indicate random mean, while red bars indicate standard deviation over 100 trials).

In order to compare the physical contact probabilities of the chromosomal regions (C_p) to the human network, we first derived a "connection tendency matrix" (M_c) at 1 Mb resolution from the human network adjacency matrix. In the M_c matrix the element in position (i, j) reports the connection tendency between genes in the i^{th} Mb and genes in the j^{th} Mb. To generate the M_c matrix, we first subdivided the adjacency matrix by grouping together probes referring to genes which where within 1 Mb distance between each other, we then computed the IS (Eq. 5.2) genome-wide between the 1 Mb regions, with K_A equal to the number of genes in the A region

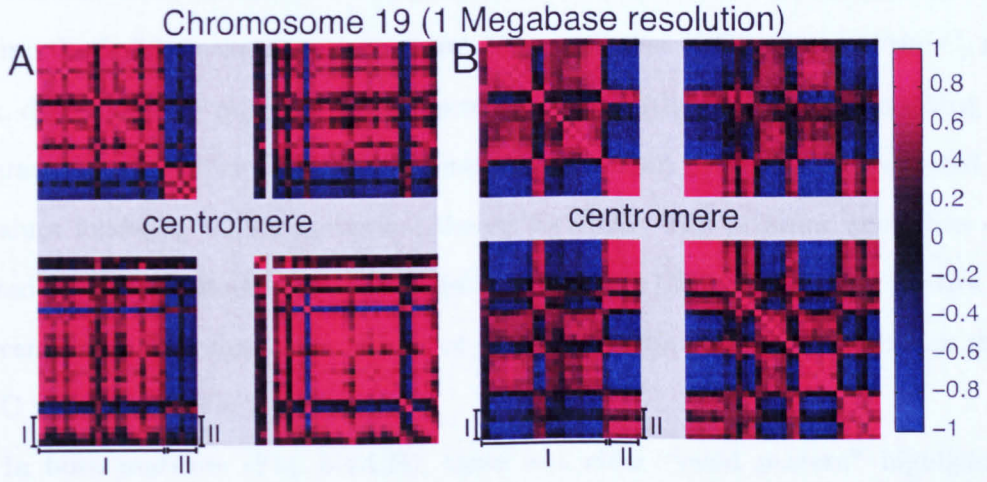


Figure 6.4: Genes that are co-expressed tend to be physically close at the 3D chromatin level. (A) Connection tendency matrix of chromosome 19. Grey stripes highlight regions with no probes designed for the microarray model HG-U133A. A red color indicates two different 1 Mb loci whose genes are strongly connected to each other. (B) Physical contact matrix of chromosome 19. Grey stripes highlights chromosomal regions where centromeres are located, plus unalignable regions. A red color indicates two different 1 Mb loci that are physically close to each at the chromatin level. Physically close regions may also contain genes that are not co-expressed and vice-versa: region (I) in (A) has an opposite tendency with respect to the corresponding region (I) in (B). This means that regions that are not in physical contact may contain genes that are co-expressed. The opposite can also be true, for example region (II) shows that loci physically interacting with each other do not necessarily contain genes that are co-expressed.

(of 1 Mb), K_B the number of genes in the B region (of 1 Mb), n_{AB} the number of interactions between genes in region A and genes in region B, and f the average frequency of interactions across all the genes in the human network. We then derived a correlation matrix C_c as shown in Figure 6.4A, where the element in position (i, j) is the Pearson correlation between the i^{th} row and j^{th} column of the M_c matrix.

Chromosome-wise analysis and genome wide analysis was accomplished by computing the 2-dimensional Pearson correlation coefficient (PCC) between the C_p matrix, describing the physical contact probability, and the C_c matrix, describing the interaction probabilities among the genes in the human network. We computed the p-values following classic statistical theory for PCC. The p-values were then corrected following a Bonferroni False Discovery Rate (FDR) procedure to control the percent of false predictions in the set of predictions. We deemed as significant those PCC with an $FDR < 0.05$.

In both matrices (Fig. 6.4A,B), there is a clear "plaid pattern" highlighting chromosomal regions whose genes are strongly connected to each other (red in Fig. 6.4A) and regions which are physically close to each other at the chromatin level (red in Fig. 6.4B). These regions have a striking overlap (correlation = 0.4, $P = 7.3 \times 10^{-123}$) especially the p-arm of chromosome 19 (upper left square in both matrices), revealing that genes that are physically close to each other at the chromatin level tend to be "co-expressed" (i.e. have a significant MI), and vice-versa.

By extending this analysis to all of the chromosomes, we found a significant overlap (correlation significance: $P < 0.01$) for all but three chromosomes (9, 20 and 21).

species	K	N	N_int_K	CORRECT	PERC	CLASS
human	17174	16750	13123	7626	58%	BP
	18141	16646	13687	6601	48%	MF
	18020	17561	14400	10247	71%	CC
mouse	19011	24848	11379	6682	59%	BP
	22774	24796	13407	6789	51%	MF
	21975	24942	13338	10375	78%	CC

Table 6.2: GOEA: guilty-by-associations study. The CLASS column reports the percentage of correct predictions; *probeset-GO term* correct association by looking at the probeset neighborhoods. K: probesets associated to one or more GO term; N: probesets whose neighborhood is associated to one or more GO term; K_int_N: intersection between K and N; K_int_N_correct: probesets associated to one or more GO term that is enriched in its neighborhood.

6.3 Prediction of gene function and protein localisation

We exploited the information embedded in the human and mouse networks to identify gene function, or protein subcellular localisation, via a *guilty-by-association* analysis. It consists in assigning a function to a gene (or a localisation to the encoded protein) by checking whether there is a shared function among the genes connected to it (or a shared localisation of their protein products). In what follows, we term "gene neighbours" the set of genes connected to a given gene of interest according to the predicted networks of connections.

We performed a Gene Ontology Enrichment Analysis (GOEA) on the set of gene neighbours for each gene, in both the human and mouse networks[1] (see Sec. 6.3.1). We then selected, as a test set, the subset of 18,141 human and 22,774 mouse transcripts for which the function/localisation was known according to their Gene Ontology classification. In table 6.2 we report the percentage of correct predictions, for each of the three GO classes (Biological Process, Molecular Function and Cellular Localisation), ranging from 48% to 78%.

[1] http://netview.tigem.it

6.3.1 Computational methods

Gene Ontology Enrichment Analysis (GOEA) is a commonly used technique that allows the identification of statistically over-represented Gene Ontology terms in a set of genes. Suppose to have a set of N genes, m of which are annotated as associated to a Gene Ontology term of interest. Suppose we draw a subset of n genes from the complete list of N genes, then the probability of obtaining k genes all sharing the same Gene Ontology term of interest follows an hypergeometric probability distribution, that is possible to compute applying the formula presented in Section 5.4. From this, we can compute the cumulative distribution and hence the significance, or p-value, of the draw; i.e. the probability of having at least k genes sharing the same GO term of interest.

In order to predict gene function and/or localisation from the human and mouse networks, we proceeded as follows: for each probe (i.e. gene) of the human or mouse network, we selected the probes predicted to be connected with it in the network (i.e. the gene's neighbours). For probes with more than 500 neighbors, only the top ranked 500 ones with the highest MI were retained; for nodes with less than 50 neighbors, we included also the gene's second neighbours (i.e. the neighbours of the neighbours) up to a maximum of 500.

Multiple probes may refer to the same gene and thus be highly co-expressed. To avoid biases in the GOEA computation, we removed from the neighbours, the probes associated to the same gene. GOEA was then performed on this subset of neighbours for each of the probe in the human network. In the testing phase, a prediction was claimed to be correct if the GO term that was enriched within a neighbourhood was also the one associated to the probe itself according to the GO database [6].

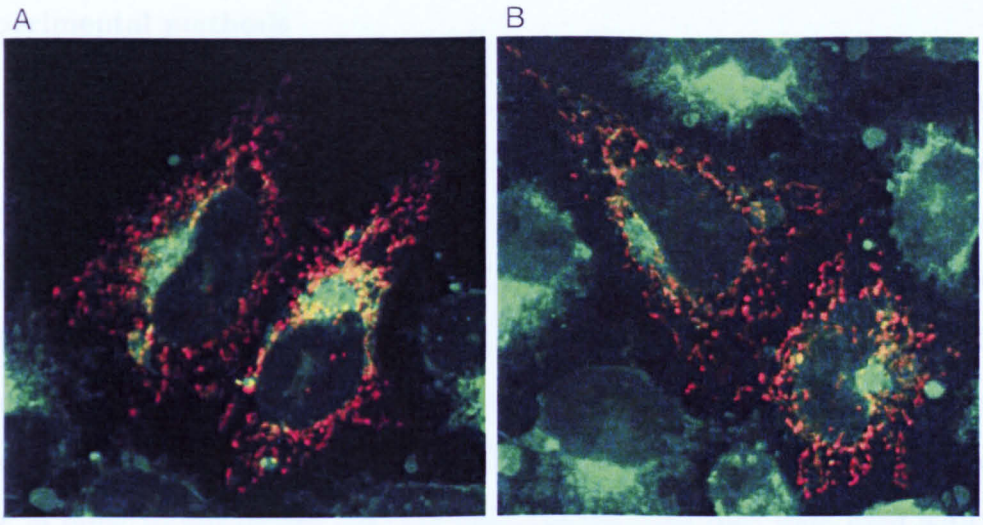


Figure 6.5: Biological validation of predicted mitochondrion protein localisations. NSUN4 (A) and UQCC (B) have been modified in order to be expressed together with a red-fluorescent protein. Green fluorescence localises mitochondrion. Yellow spot are given by the combination of red and green showing the mitochondrion localisation of the expression of the protein we are testing.

6.3.2 Biological validation of gene expression localisation

To test experimentally a subset of new protein localisation predictions, we selected 6 genes, whose protein products were predicted to be localised in mitochondria by the guilty-by-association analysis (Sec. 6.3.1), but that, to our knowledge, were not yet reported in the literature. We fused the corresponding cDNA to the N-terminal of a stabilised green fluorescent protein (EGFP), and transfected HeLa cells with these constructs.

We found that 2 genes, NSUN4 (*NOP2/Sun domain family, member 4*) and UQCC (*ubiquinol-cytochrome c reductase complex chaperone, CBP3 homolog (yeast)*) localised in mitochondria as shown in Figure 6.5); confirming that at least 33% of our predictions are correct. For comparison, had 6 genes been selected at random, the probabiltiy that 2 of them had protein products localised to mitochondria would have been less that 0.3%, hence two-order of magnitudes lower.

Experimental methods

Cell culture and transfection for protein mitochodrial localisation assays. HeLa cells were maintained at 37°C in a 5% CO2-humidified incubator, and cultured in DMEM (GIBCO BRL) supplemented with 10% heat-inactivated fetal bovine serum (FBS) (Invitrogen) and 1% antibiotic-antimycotic solution (GIBCO BRL). 50000 cells were transfected in a 24-well with 0.3μg of DNA using TransIT-LT1 Transfection Reagent following the standard protocol suggested by the manufacturer. After 16h cells were treated with a red MitoTracker (Invitrogen) for 30 minutes, in order to specifically bind the mitochondria, and then were fixed with 4% paraformaldehyde for 10 minutes. Cells were observed using the confocal microscope ZEISS LSM 710.

Bacterial strain, Plasmids and Molecular Cloning. Plasmid construction and molecular cloning were performed in the cloning host cell E.coli DH5a (Invitrogen) following standard protocols. For each gene of interest the coding sequence was amplified from a total preparation of cDNA obtained by RT-PCR of a murine RNA.

The amplification was carried out using the primers listed in Table 6.3. Each coding sequence was then cloned into the EcoRI/BglII sites (except for Fam73b that was cloned into EcoRI/BamHI sites) of pEGFP-N3 vector at the N terminal of the EGFP, thus resulting in a fusion protein. All clones were confirmed by sequencing analysis.

6.4 Elucidating the function of the granulin precursor 'disease-gene'

We next asked whether the guilty-by-association approach could be helpful in elucidating the function of genes involved in genetic diseases. Since we were interested

Gene	Sequence
9530058B02Rik	F (5'-CAGGAATTCACCATGTACACTATCACCAAG -3')
9530058B02Rik	R (5'-CGGAGATCTGGAACAGTTGGTGATTTTAGC-3')
2010012O05Rik	F (5'-TTAGAATTCATGATGGCGACTGGGACGC-3')
2010012O05Rik	R (5'-GCGAGATCTCTTCAGTAAGTGACTCAACTG-3')
2810405K02Rik	F (5'-GTCGAATTCATTATGAATGTGGTGGACCTTGGT-3')
2810405K02Rik	R (5'-GCCAGATCTCCTCCCACACACCTCTTCA-3')
2010012O05Rik	F (5'-GCGGAATTCAGGTGATGGCGACTGGGAC-3')
2010012O05Rik	R (5'-GTTAGATCTCTTCAGTAAGTGACTCAACTGGTC-3')
Nsun4	F (5'-CTAGAATTCACCATGGTGTTTATCACATCAATAGA-3')
Nsun4	F (5'-CTAGAATTCACCATGGCTGCGCCCGTCTTAA-3')
Nsun4	R (5'-CGCAGATCTTGGCAGCCTATGCAGTTTGC-3')
Uqcc	F (5'-GCGGAATTCAGTATGGCGTTGCTGGTGCGAG-3')
Uqcc	F (5'-GCGGAATTCAAGATGGGATTCACTGGACCTTTG-3')
Uqcc	R (5'-CGAAGATCTAAGGCCCTCATCATTGTAAGTA-3')
Ccbl1	F (5'-GCCGAATTCACCATGTCCAAACAGCTGCAGGC-3')
Ccbl1	R (5'-CGCAGATCTGGCTTGGGGCTCTCCTTTC-3')

Table 6.3: Genes tested for mitochondrial localisation. Genes predicted to be significantly associates to mitochondria by the guilty-by-association analysis. The sequence column reports the primers used to test the mitochondrial localisation. Names in italics indicate an alternative isoform of the protein. F and R state for Forward and Reverse, respectively.

in lysosomal function and lysosomal disorders, we used our guilty-by-association analysis to identify human disease genes which may have a yet undiscovered role in lysosome function and organization.

To this end, we ranked in descending order of their p-values all of the genes predicted to be lysosomal, or associated with *lysosome organization*, by our guilty-by-association analysis (http://netview.tigem.it). The top ranked genes included both lysosomal enzymes and other genes involved lysosomal function and are listed in Table 6.4 and Table 6.5. Among these, at position one, *NPC2* (*Niemann-Pick disease, type C2*) disease-gene, known to be an intralysosomal gene [116]; at position two, we found another disease-gene *GRN*. Both genes are members of community 40, which is enriched both the presence of disease genes and for lysosomal genes, described in Section 5.4.

Despite recent extensive studies, the role of *GRN* is far from being understood and, to our knowledge, it has not been directly linked to lysosomal function in the literature. *GRN* is a highly conserved gene bearing multiple copies of the cysteine-

P-value	Probeset ID	Gene Symbol	Gene Info
7.89×10^{-41}	200701_at	NPC2	Niemann-Pick disease, type C2
1.88×10^{-36}	211284_s_at	GRN	granulin
2.41×10^{-34}	200743_s_at	TPP1	tripeptidyl peptidase I
7.74×10^{-33}	212663_at	FKBP15	FK506 binding protein 15, 133kDa
3.76×10^{-32}	201212_at	LGMN	legumain
7.10×10^{-31}	201494_at	PRCP	prolylcarboxypeptidase (angiotensinase C)
8.76×10^{-31}	218217_at	SCPEP1	serine carboxypeptidase 1
5.99×10^{-30}	202944_at	NAGA	N-acetylgalactosaminidase, alpha
8.99×10^{-30}	217118_s_at	C22ORF9	chromosome 22 open reading frame 9
9.02×10^{-30}	204204_at	SLC31A2	solute carrier family 31, member 2
2.31×10^{-29}	200871_s_at	PSAP	prosaposin
9.49×10^{-29}	201944_at	HEXB	hexosaminidase B (beta polypeptide)
2.73×10^{-28}	35820_at	GM2A	GM2 ganglioside activator
3.67×10^{-28}	200839_s_at	CTSB	cathepsin B
7.29×10^{-28}	202295_s_at	CTSH	cathepsin H
7.29×10^{-28}	202838_at	FUCA1	fucosidase, alpha-L- 1, tissue
8.89×10^{-28}	200748_s_at	FTH1	ferritin, heavy polypeptide 1
5.39×10^{-27}	202545_at	PRKCD	protein kinase C, delta
7.00×10^{-27}	202087_s_at	CTSL1	cathepsin L1
8.61×10^{-27}	208704_x_at	APLP2	amyloid beta (A4) precursor-like protein 2

Table 6.4: Top 20 genes significantly associated to *Lysosome* (cellular component gene ontology class) resulting from the application of the guilty-by-association analysis, sorted according to the associated p-values.

P-value	Probeset ID	Gene Symbol	Gene Info
1.58×10^{-09}	200661_at	CTSA	cathepsin A
7.17×10^{-06}	200742_s_at	TPP1	tripeptidyl peptidase I
3.20×10^{-05}	207809_s_at	ATP6AP1	ATPase, H+ transporting, lysosomal accessory protein 1
3.48×10^{-05}	211284_s_at	GRN	granulin
2.10×10^{-04}	202545_at	PRKCD	protein kinase C, delta
2.25×10^{-04}	201050_at	PLD3	phospholipase D family, member 3
2.90×10^{-04}	200766_at	CTSD	cathepsin D
3.73×10^{-04}	217118_s_at	C22ORF9	chromosome 22 open reading frame 9
3.74×10^{-04}	219952_s_at	MCOLN1	mucolipin 1
4.58×10^{-04}	202812_at	GAA	glucosidase, alpha;
2.54×10^{-03}	208926_at	NEU1	sialidase 1 (lysosomal sialidase)
3.54×10^{-03}	205090_s_at	NAGPA	N-acetylglucosamine-1-phosphodiester alpha-N-acetylglucosaminidase
3.81×10^{-03}	200649_at	NUCB1	nucleobindin 1
4.56×10^{-03}	218282_at	EDEM2	ER degradation enhancer, mannosidase alpha-like 2
4.56×10^{-03}	219020_at	HS1BP3	HCLS1 binding protein 3
4.56×10^{-03}	203045_at	NINJ1	*ninjurin 1*
4.56×10^{-03}	201494_at	PRCP	prolylcarboxypeptidase (angiotensinase C)
4.56×10^{-03}	212647_at	RRAS	related RAS viral (r-ras) oncogene homolog
4.56×10^{-03}	218217_at	SCPEP1	serine carboxypeptidase 1
4.56×10^{-03}	203167_at	TIMP2	TIMP metallopeptidase inhibitor 2

Table 6.5: Top 20 genes significantly associated to *Lysosome organisation* (biological process gene ontology class) resulting from the application of the guilty-by-association analysis, sorted according to the associated p-values.

rich granulin motifs. Proteolytic cleavage of the precursor protein by extracellular proteases, such as elastase, gives rise to smaller peptide fragments termed granulins, or epithelins, which have been linked to a range of biological functions including cell division, survival, and migration [37]. Mutations in *GRN* cause frontotemporal lobar degeneration with ubiquitin-immunoreactive neuronal inclusions (FTLD-U) [7, 27]. Of note, mutations in *NPC2* results in a wide spectrum of clinical phenotypes including a form of frontal lobe atrophy [64].

To investigate whether *GRN* was indeed related to lysosomal function, we first evaluated *GRN* expression levels following sucrose treatment, a known inducer of lysosomal biogenesis [62, 52]. A strong increase in the number of lysosomes was detected upon sucrose supplementation to the culture medium, as expected (Data not shown). Following sucrose treatment, in Figure 6.6A, we observed a 2-fold increase over baseline in *GRN* mRNA levels, along with a 3-fold increase in Cathepsin D (CTSD), a lysosomal enzyme used as positive control (Figure 6.6A).

The transcription factor EB (*TFEB*) has been recently identified as the transcription factor controlling most of the known lysosomal genes via direct binding to their proximal promoter [96], therefore we next asked whether *GRN* may indeed be regulated by this transcription factor. We first identified, by bioinformatics analysis, two *TFEB* binding sites upstream of the *GRN* coding sequence (see section 6.4.2). We then over-expressed *TFEB* in human cell lines and detected a 3-fold increase in *GRN* mRNA levels, along with a 3-fold increase in *CTSD*, a known target of *TFEB*, used as a positive control (Figure 6.6B).

We next over-expressed *GRN* to observe if this had any effect on lysosomes; as shown in Figure 6.6C, the number of lysosomes significantly increased ($P = 0.01$), compared to a mock control, or over-expression of EGFP, as assessed by immunofluorescence against LAMP2. The effect of increased lysosomes was observed on the great majority of cells. Therefore, we hypothesised that *GRN*-transfected cells could secrete a factor inducing lysosome biogenesis in neighbouring cells. This hypothesis

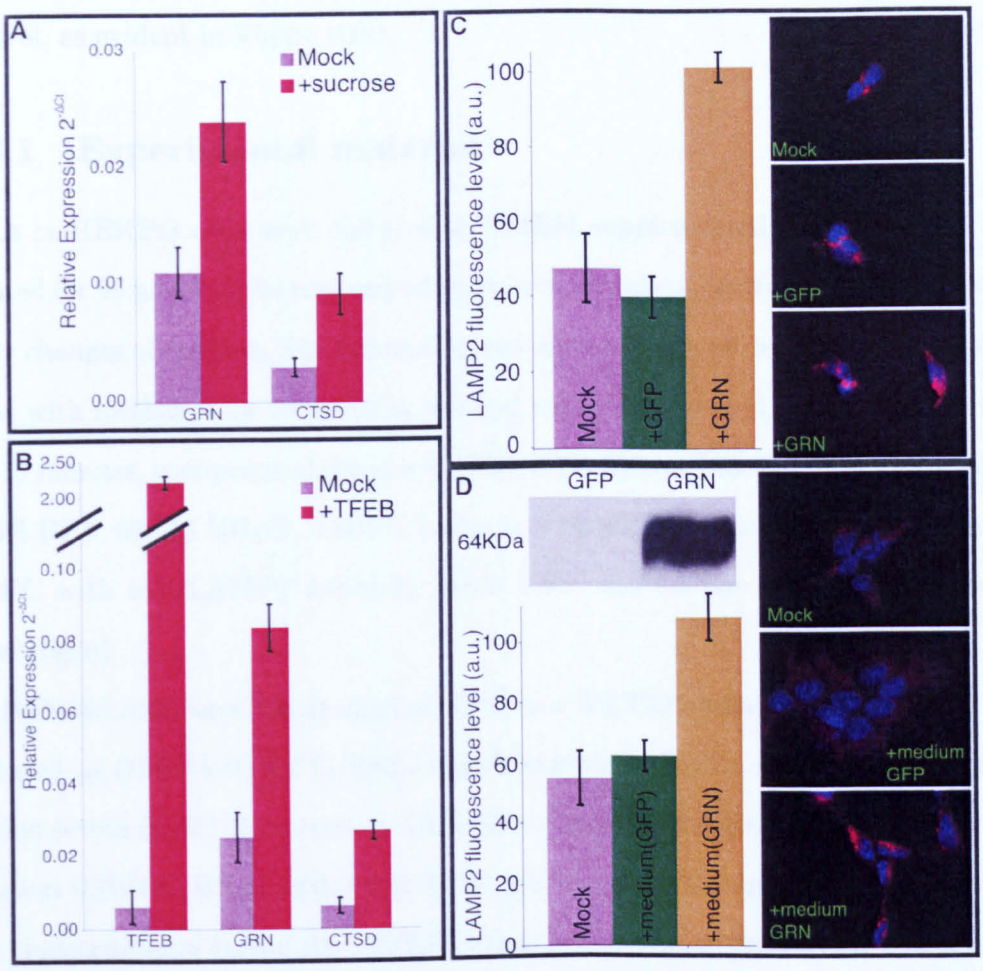


Figure 6.6: *GRN* is involved in lysosomal biogenesis and function. (A) *GRN* and *CTSD* increase in expression level following sucrose treatment, as measured by realtime PCR. (B) Expression level of *GRN* and of *CTSD* increase following *TFEB* over-expression. (C) Immuno-fluorescence with antibody anti-*LAMP2*, used as a lysosomal marker, of transfected HEK293 cells over-expressing *GRN*, or EGFP, against a mock control. All values represent mean fluorescence intensity ±s.e.m; *GRN* overexpression ($P = 0.003$) and sucrose treatment ($P = 0.01$) significantly increase the fluorescence compared to mock control. (D) Immuno-fluorescence with antibody anti-*LAMP2* in wt HEK293 grown in medium collected from *GRN* over-expressing cells shows a significant increase in fluorescence ($P = 0.008$) compared to medium collected from EGFP over-expressing cells.

was supported by a previous study indicating that *GRN* is uptaken by a mannose 6P-phospate receptor mechanism [65]. To test this hypothesis, we collected the medium from transfected cells over-expressing *GRN*, and used it to grow untreated cells; this resulted in a consistent increase ($P = 0.003$) in lysosome number compared to

control, as evident in Figure 6.6D.

6.4.1 Experimental material

HeLa or HEK293 cells were cultured in DMEM supplemented with 10% FBS and treated for 96 hours in the presence of sucrose to a final concentration of 100mM with daily changes of medium. For immunofluorescence cells grown on glass coverlips were fixed with methanol for 10 minutes, washed with PBS, treated with 50 mM NH_4Cl for 15 minutes, and permeabilized with PBS 0.1% Triton, blocked in blocking buffer (0.5% BSA, 50mM NH_4Cl, 0.001% Triton in PBS pH 7.4), and incubated overnight at 4°C with anti-LAMP2 antibody Santa Cruz and for one hour with Alexa-594 (Invitrogen).

HEK293 cells were maintained at 37°C in a 5% CO2-humidified incubator, and cultured in DMEM (GIBCO BRL) supplemented with 10% heat-inactivated fetal bovine serum (FBS) (Invitrogen), 1% L-glutammine and 1% antibiotic/antimycotic solution (GIBCO BRL). 500,000 cells were trasfected with 4μg of DNA expressing the transcriptional factor EB (TFEB) using lipofectamine trasfection reagent (Invitrogen) following the standard protocol suggested by the manufacturer. After 48 hours from trasfection cells were collected, the mRNA extracted and the levels of Granulin and Catepsin D (CTSD) were evaluated by Real-Time PCR (Roche). The amplification was performed using the following primers for GAPDH, "Fw: GAAGGTGAAGGTCGGAGTC" and "Rev: GAAGATGGTGATGGGATTTC", for GRN, "Fw: TCCAGAGTAAGTGCCTCTCCA" and "Rev: TCACCTCCATGTCACATTTCA", and for CTSD, "Fw: AACTGCTGGACATCGCTTGCT" and "Rev: CATTCTTCACGTAGGTGCTGGA".

Medium was concentrated on filters (Vivaspin Sartorius Stedim) and loaded on 10% SDS-PAGE. Transfer membranes were incubated with anti human granulin antibody (Invitrogen) at 1:50 dilution.

6.4.2 Identification of Binding sites in Granulin promoter region

We used the Position Weight Matrix of the Transcription Factor EB and run MATCH [63] to find its binding sites across -1000 +1000 base pairs flanking the UTR region of GRN. We found two binding sites on chromosome 17, plus strain, with location 42,422,444 to 42,422,457 and chromosome 17, plus strain, location 42,422,460 42,422,473 (Genome Browser, Human assembly Feb. 2009).

6.5 Gene signature analysis

The human network can be used as a 'consensus' network to analyse gene signatures. A gene signature is a list of genes able to identify a specific condition (i.e. disease state, drug response, etc.). Genes part of a signature are the end point of a regulatory cascade. The identification of the primary genes, such as Transcription Factors (TFs), that control the genes in the signature via the regulatory cascade can help in identifying the "master regulators", involved in the process of interest [24].

The Gene Signature Analysis is basically a Gene Set Enrichment Analysis and consists in computing the statistical significance of the intersection of two sets of objects. In this case, we check how significant the group of genes in the gene signature of interest are represented within the neighbours of each TF of the human network.

6.5.1 A case study

As an example of the use of the network, we analysed the genes part of the mesenchymal gene expression signature (MGES) found to be over-expressed in poor prognosis group of glioma patients [89]. In [24], a high-grade glioma (HGG) specific gene network was inferred from 176 expression profiles of HGG samples derived from patients. This condition-specific network was then used to infer the TFs responsible

Common	number of neighbours	p-value	Gene Symbol
8	35	1.33×10^{-11}	MAFF
10	93	9.17×10^{-11}	AEBP1
7	43	3.39×10^{-09}	FOSL2
5	12	3.48×10^{-09}	FOSL1
7	53	1.55×10^{-08}	ELF4
7	71	1.23×10^{-07}	TFE3
7	86	4.66×10^{-07}	JUNB
10	229	5.45×10^{-07}	IRF1
6	55	5.54×10^{-07}	RELB
6	65	1.51×10^{-06}	CEBPB
5	73	5.05×10^{-05}	TNXA
4	40	6.71×10^{-05}	BHLHB2
5	96	1.86×10^{-04}	RELA
3	26	3.80×10^{-04}	CEBPD
10	595	1.62×10^{-03}	PRRX1
4	96	1.93×10^{-03}	BTG2
3	56	3.59×10^{-03}	NFKB1
2	18	4.30×10^{-03}	LOC161527
3	65	5.46×10^{-03}	SMAD3
4	136	6.69×10^{-03}	IRF7
2	23	6.98×10^{-03}	RARA
4	142	7.77×10^{-03}	SPI1
4	152	9.82×10^{-03}	STAT3
3	85	1.14×10^{-02}	PYCARD
2	31	1.25×10^{-02}	CREB3L1
2	40	2.02×10^{-02}	FOSB
3	113	2.43×10^{-02}	JUND
2	46	2.63×10^{-02}	SNF1LK
2	48	2.85×10^{-02}	HMOX1
3	131	3.54×10^{-02}	MYD88
2	61	4.41×10^{-02}	ATF3

Table 6.6: Transcription Factors that are enriched of gene in the mesenchymal gene expression signature. MGES accounts 122 probes. Genes are ranked in ascending order according with their p-values.

for the mesenchymal transformation of brain tumors. Such "master regulators" were the TFs that, according to the HGG-specific transcriptional network, regulated a significant number of genes in the MGES signature.

We identified 31 TFs listed in Table 6.6 that according to our network had a significant number of connections to the genes in the signature. Eighteen TFs, out of these 31, were identified also in [24] (significance of $p = 7.3 \times 10^{-16}$). The two key genes, C/EBPβ and STAT3, codifying for TFs necessary in human glioma cells for mesenchymal transformation [24] were correctly included among the 31 TFs.

We further tested whether the genes in the MGES signature were over-represented within some of the 393 human communities identified within the network. Community 11 and 46 were found to be significantly enriched ($p = 7.90 \times 10^{-16}$ and $p = 1.09 \times 10^{-12}$, respectively) for genes in the MGES signature. Community 11 is composed by genes involved in *phosphate transport* ($p = 5.35 \times 10^{-38}$), and whose expression is localised into the *proteinaceous extracellular matrix* ($p = 1.70 \times 10^{-99}$); whereas community 46 includes genes involved in *actin filament-based process* ($p = 1.50 \times 10^{-06}$), and expressed into the *cytoskeleton* ($p = 6.23 \times 10^{-10}$).

6.6 Discussions and conclusions

We demonstrated that genes connected within functional modules in the human network tend to have a 'physical counterpart' in the three-dimensional conformation of the chromatin inside the nucleus. We observed a striking similarity between genes that appear to be connected, and, therefore, are co-expressed, and their physical proximity at the three-dimensional chromatin level. This suggests that regulation of coordinated gene expression is "hard-wired" in the physical arrangement of the chromatin within the nucleus.

We have shown, and experimentally validated, different examples of how the network can be queried to predict the the localisation of genes'protein product, to identify new protein-protein interactions and to explore the function of a disease-gene.

Upregulation of *GRN* by known inducers of lysosomal biogenesis and function, together with the increase in the number of lysosomes following *GRN* overexpression, or treatment with medium from over-expressing cells, clearly supports a role of *GRN* in lysosome biology that has been unrecognized until now. This finding is also supported by previous evidence indicating that *GRN* colocalises with lysosome-associated CD68 antigen in activated macrophages and microglia [81] and is overexpressed in the cerebral cortex of MPSIIIB and MPSI mice [85].

Moreover, *GRN* has been shown to bind the mannose 6-phosphate receptor [65]. Interestingly, lysosomal dysfunction has been proposed to play a role in other neurodegenerative diseases including Alzheimer's disease [25] sharing clinical similarities with FTLD-U. In these disorders it has been proposed that lysosomal hydrolytic enzymes contribute to cell death through lysosomal destabilization and enzyme leakage into the cytoplasm as indicated by models of experimental brain ischemia injury resulting in cytosolic acidification and rupture/permeabilization of lysosomes [117]. Interestingly, one of the first recognized function of *GRN* has been in wound healing [51] and release of lysosomal enzymes appears to play an important role in healing [61, 78].

The Gene Signature Analysis we proposed in Section 6.5, shows that it may be not always necessary to use a diseases-specific network to analyse a disease-specific gene signature, but using a "consensus" network, we could correctly identify the "master regulators" involved in the observed gene signature.

Chapter 7

Conclusions and future directions

Despite the common belief that massive and heterogeneous gene expression profiles would be too noisy to be used for inference, in this thesis work we have demonstrated the biological reliability of the human and mouse gene regulatory networks we inferred. The inferred networks were compared with known protein-protein and other types of interactions collected from literature. We also experimentally confirmed different types of predictions (protein-protein, gene function and gene localisation). Moreover, we discovered topological properties of the inferred networks, such as: degree distribution, small protein disorder associated to hubs, modularity structure of the networks and the presence of co-expressed groups of genes that localise in spatially close chromosomal loci. However, the heterogeneity of the data could bias the analysis toward the most representative biological condition. The availability of a more precise annotation of expression data will facilitate the use of this as well as of similar procedures.

The results of this work can also be explored online[1]. The online tool provides both the access to the inferred connections of the human and mouse networks, and to a set of gene function predictions obtained by analysing the topology of the network with the "guilty-by-association" approach presented in Chapter 6.

[1] http://netview.tigem.it

The computational core of our information-theoretic approach is the mutual information (MI). The MI measures the statistical dependence between two variables, in general, and two genes in the particular case of gene network inference. The algorithm reported in Chapter 4 is able to compute the point estimate of the MI for each pair of genes in the mammals species we considered. The point estimate however does not allow to estimate the confidence interval for the MI. Here, we propose a statistical model that theoretically could be used to obtain the confidence interval for the MI associated to a pair of genes.

7.1 Estimation of MI via a hierarchical statistical model

Suppose to have K experiments. Each experiment is a collection of hybridisations, hence a collection of gene expression levels. Suppose to discretise the expression levels in a pre-determined number of classes C, by following the discretisation procedure reported in Section 4.2.1. For a pair of genes, or variables in general, for each experiment we have a vector of counts that explains how coherently the expression levels vary together along the set of hybridisations. We remind the these counts or outcomes were directly used in Chapter 4 to compute the MI between pairs of genes. In that case, we assumed that the marginals and joint probability could be well approximated by the frequencies of the outcomes. Here the vectors of counts are not used directly, but they are assumed to follow an independent multinomial distribution across the experiments $k = 1, ..., K$,

$$\mathbf{n}_k \sim Mult(M_k, \boldsymbol{\theta}_k) \tag{7.1}$$

where M_k is the number of hybridisations. The number of outcomes that may be observed is equal to C^2. The parameters $\boldsymbol{\theta}_k$ are unknown and they are assumed to

be independent samples from a Dirichlet distribution,

$$\boldsymbol{\theta}_k \sim Dirichlet(\boldsymbol{\alpha}). \tag{7.2}$$

We shall also assign a noninformative hyperprior distribution to reflect our ignorance about the unknown hyperparameters

$$\boldsymbol{\alpha} \sim \prod_{j=1}^{J} Gamma\left(\alpha_j \mid a, b\right). \tag{7.3}$$

where J is the number of dimension of the parameter vector $\boldsymbol{\alpha}$, as well as, $\boldsymbol{\theta}_k$ and $\mathbf{n}_k$. The hierarchical model just described was presented in [45], using as conjugate distributions the Binomial (instead of the Multinomial) and the Beta (instead of the Dirichlet).

7.1.1 Joint, conditional and marginal posterior distributions

We can first perform the three steps for determining the analytic form of the posterior distribution of $\boldsymbol{\alpha}$ and $\boldsymbol{\theta}_k$s parameter vectors. The joint posterior distribution, $p(\boldsymbol{\alpha}, \boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_K \mid \mathbf{n}_1, ..., \mathbf{n}_K)$, of all parameters is

$$\propto \quad p(\boldsymbol{\alpha})p(\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_K \mid \boldsymbol{\alpha})p(\mathbf{n}_1, ..., \mathbf{n}_K \mid \boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_K, \boldsymbol{\alpha}) \tag{7.4}$$

$$\propto \quad p(\boldsymbol{\alpha}) \prod_{k=1}^{K} \left[Dirichlet(\boldsymbol{\theta}_k \mid \boldsymbol{\alpha}) Mult(\mathbf{n}_k \mid M_k, \boldsymbol{\theta}_k)\right] \tag{7.5}$$

$$= \quad p(\boldsymbol{\alpha}) \prod_{k=1}^{K} \left[\frac{\Gamma(\sum_{j=1}^{J} \alpha_j)}{\prod_{j=1}^{J} \Gamma(\alpha_j)} \prod_{j=1}^{J} \theta_{kj}^{\alpha_j - 1} \frac{\mathbf{n}_k!}{\prod_{j=1}^{J} n_{kj}!} \prod_{j=1}^{J} \theta_{kj}^{n_{kj}}\right]$$

$$= \quad p(\boldsymbol{\alpha}) \prod_{k=1}^{K} \left[\frac{\Gamma(\sum_{j=1}^{J} \alpha_j)}{\prod_{j=1}^{J} \Gamma(\alpha_j)} \cdot c'_k \cdot \prod_{j=1}^{J} \theta_{kj}^{n_{kj} + \alpha_j - 1}\right], \tag{7.6}$$

with $c'_k = \frac{\mathbf{n}_k!}{\prod_{j=1}^{J} n_{kj}!}$. Notice that (7.5) follows (7.4) because $\mathbf{n}_k$ is independent from $\boldsymbol{\alpha}$ given $\boldsymbol{\theta}_k$, for $k = 1, ..., K$.

Given $\boldsymbol{\alpha}$, all the $\boldsymbol{\theta}_k$'s have independent posterior densities given by

$$p(\boldsymbol{\theta}_k \mid \boldsymbol{\alpha}, \mathbf{n}_k) \propto \prod_{j=1}^{J} \theta_{kj}^{n_{kj}+\alpha_j-1} \sim Dirichlet(\boldsymbol{\alpha} + \mathbf{n}_k), \tag{7.7}$$

and the joint density is

$$p(\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_K \mid \boldsymbol{\alpha}, \mathbf{n}_1, ..., \mathbf{n}_K) \propto \prod_{k=1}^{K} \left[\prod_{j=1}^{J} \theta_{kj}^{n_{kj}+\alpha_j-1} \right]. \tag{7.8}$$

From (7.6), integrating over the $\boldsymbol{\theta}$'s, we can determine the marginal posterior distribution of $\boldsymbol{\alpha}$

$$p(\boldsymbol{\alpha} \mid \mathbf{n}_1, ..., \mathbf{n}_K) \propto \tag{7.9}$$

$$\propto \quad p(\boldsymbol{\alpha}) \prod_{k=1}^{K} \left[c' \cdot \frac{\Gamma(\sum_{j=1}^{J} \alpha_j)}{\prod_{j=1}^{J} \Gamma(\alpha_j)} \int_{\boldsymbol{\theta}_k} \prod_{j=1}^{J} \theta_{kj}^{n_{kj}+\alpha_j-1} d\boldsymbol{\theta}_k \right]$$

$$= \quad p(\boldsymbol{\alpha}) \cdot c'' \cdot \prod_{k=1}^{K} \frac{\Gamma(\sum_{j=1}^{J} \alpha_j)}{\prod_{j=1}^{J} \Gamma(\alpha_j)} \frac{\prod_{j=1}^{J} \Gamma(n_{kj}+\alpha_j)}{\Gamma(\sum_{j=1}^{J} n_{kj}+\alpha_j)} \cdot \int_{\boldsymbol{\theta}_k} \frac{\Gamma(\sum_{j=1}^{J} n_{kj}+\alpha_j)}{\prod_{j=1}^{J} \Gamma(n_{kj}+\alpha_j)} \prod_{j=1}^{J} \theta_{kj}^{n_{kj}+\alpha_j-1} d\boldsymbol{\theta}_k \tag{7.10}$$

where the last factor of Equation 7.10 integrates to 1 for $k = 1, ..., K$ given that $\int_{\theta} p(\theta \mid \alpha) d\theta = 1$; $c'' = \prod_{k=1}^{K} c'_k$. The posterior distribution of the hyper-parameters becomes

$$\propto \quad p(\boldsymbol{\alpha}) \prod_{k=1}^{K} \left[\frac{\prod_{j=1}^{J} \left[(\alpha_j + n_{kj}) \cdot (\alpha_j + n_{kj} - 1) \cdot ... \cdot (\alpha_j + 1) \right]}{\left(\sum_{j=1}^{J} \alpha_j + M_k \right) \cdot \left(\sum_{j=1}^{J} \alpha_j + M_k - 1 \right) \cdot ... \cdot \left(\sum_{j=1}^{J} \alpha_j + 1 \right)} \right] \tag{7.11}$$

$$= \quad p(\boldsymbol{\alpha}) \prod_{k=1}^{K} \left[\frac{\prod_{j=1}^{J} [\alpha_j]_{n_{kj}}}{\left[\sum_{j=1}^{J} \alpha_j \right]_{M_k}} \right] \tag{7.12}$$

where Equation 7.11 follows the Equation 7.12 by applying a well known property of the Gamma function that states that $\Gamma(z+1) = z \cdot \Gamma(z)$.

7.1.2 Algorithm for estimation of the MI

The posterior distribution of the hyper-paramenter vector $\boldsymbol{\alpha}$ embeds the information concerning the joint probability distribution of two genes or variables. Given the posterior distribution of the $\boldsymbol{\alpha}$ vector of hyper-parameters, we may use it to draw samples $\boldsymbol{\alpha}$, these in turn can be seen as Dirichlet parameters and used to draw the probability vectors in order to compute the MI between the pair of genes. These steps are formalised in Algorithm 1. Algorithm 1 allows to simulate the probability

Algorithm 1 SIMULATION(n)

Require: $p(\boldsymbol{\alpha} \mid \mathbf{n}_1, ..., \mathbf{n}_K)$.

1: $M = 100$ $I = 100$
2: **for** $m = 1$ to M **do**
3: $\quad \boldsymbol{\alpha}^{(m)} \sim p(\boldsymbol{\alpha} \mid \mathbf{n}_1, ..., \mathbf{n}_K)$
4: $\quad$ **for** $i = 1$ to I **do**
5: $\quad\quad \boldsymbol{\theta}^{(i)} \sim Dirichlet(\boldsymbol{\alpha}^{(m)})$
6: $\quad\quad mi^i = MI(\boldsymbol{\theta}^{(i)})$
7: $\quad$ **end for**
8: $\quad f^{(m)} = \frac{\sum_{h=1}^{M} mi^h}{I}$
9: **end for**
10: $MI(\boldsymbol{\theta} \mid \mathbf{n}_1, ..., \mathbf{n}_K))$

vectors and can be used to compute the MI of two variables (genes in this context). By performing the steps several times, we can estimate the distribution of the MI between two variables. Note that by performing the steps reported in Chapter 4, we can only compute the point estimate of the MI between two genes, with no information regarding the robustness of the estimation. Algorithm 1 instead, would end up with a distribution of MI values between two genes. From this distribution we could then estimate the average MI between the two genes, as well as, the confidence interval of our estimation.

7.2 The Dirichlet-multinomial/Polya distribution

In a real scenario like the one explored in Chapter 5, the dimensionality of the problem in terms of number of genes (gene pairs), is of the order of 10,000 (hundreds of millions). The parallel algorithm proposed in Chapter 4 was run on a cluster of 100 processors, and required 4 and 8 hours to reverse-engineer the human and mouse networks, respectively. If we had implemented the code listed in Algorithm 1, we would have had an unacceptable computational time (10,000 times larger). The Algorithm 1 can therefore be applied only when the number of genes is "small".

The Dirichlet-multinomial distribution is a compound distribution where $\boldsymbol{\theta}$ is drawn from a Dirichlet and then a sample of discrete outcome $\boldsymbol{n}$ is drawn from a multinomial with a probability vector $\boldsymbol{\theta}$. This model is essentially a "Polya" urn scheme, so the Dirichlet-multinomial is also called the *Polya distribution* [79]. Let n_i be the number of times the outcome was equal to i, i. e.

$$n_i = \sum_j \delta(x_j - i).$$

Then the resulting distribution over $\boldsymbol{n}$, the vector of outcomes, is

$$\begin{aligned} p(\boldsymbol{n} \mid \boldsymbol{\alpha}) &= \int_{\boldsymbol{\theta}} p(\boldsymbol{n} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \boldsymbol{\alpha}) d\boldsymbol{\theta} \\ &= \frac{\Gamma(\sum_i \alpha_i)}{\Gamma(\sum_i n_i + \alpha_i)} \prod_i \frac{\Gamma(n_i + \alpha_i)}{\Gamma(\alpha_i)}. \end{aligned}$$

This distribution is also parameterised by $\boldsymbol{\alpha}$, which can be estimated from a training set of count vectors: $D = \{\boldsymbol{n}_1, ..., \boldsymbol{n}_K\}$. The likelihood is

$$\begin{aligned} p(D \mid \alpha) &= \prod_k p(\boldsymbol{n}_k \mid \alpha) \\ &= \prod_k \left(\frac{\Gamma(\sum_i \alpha_i)}{\Gamma(n_k + \sum_i \alpha_i)} \prod_k \frac{\Gamma(n_{ki} + \alpha_i)}{\Gamma(\alpha_i)} \right). \end{aligned} \tag{7.13}$$

There are many different ways to estimate the parameter vector $\boldsymbol{\alpha}$, that maximises the likelihood in Equation 7.13. It is possible also possible to estimate the Polya mean and precision separately. By rewriting the likelihood in terms of the leave-one-out likelihood, a simple convergent fixed-point iteration to estimate the parameters is

$$\alpha_i^{new} = \alpha_i \frac{\sum_k \frac{n_{ki}}{n_{ki}-1+\alpha_i}}{\sum_k \frac{n_k}{n_k-1+\sum_i \alpha_i}},$$

until the desired convergence is reached. Other ways for estimating the Polya parameters can be found in [79].

The estimated parameter vector $\boldsymbol{\alpha}$ embeds the knowledge of the distribution of the variables, or genes, considered. This value of $\boldsymbol{\alpha}$ can be used to performe only step 4 to 6 of Algorithm 1. The resulting distribution can then be used to compute the MI between the pair of genes and give a measure of the robustness of the MI computed. Moreover, the parameter vector $\boldsymbol{\alpha}$ may be used as prior knowledge for future inferences, when more data come available.

Appendix A

Parameters setting for the reverse-engineering algorithm presented in Chapter 3

NIR

	Connectivity
For small gene networks of order 10:	5
For medium gene networks of order 100:	
TopD	5
Reest-K	10

Clustering (Hierarchal)

	No of Clusters
For small gene networks of order 10:	3
For medium gene networks of order 100:	10
For big gene networks of order 1000:	100

ARACNe

DPI:	0.15

BANJO

searcherChoice:	SimAnneal
initialTemperature	1000
coolingFactor	0.9
reannealingTemperature	500
maxAcceptedNetworkBeforeCooling	1000
maxProposedNetworkBeforeCooling	10000
minAcceptedNetworkBeforeReannealing	200
proposerChoice:	RandomLocalMove
evaluatorChoice:	default
deciderChoice:	default
discretizationPolicy:	Q3
minMarkovLag (for dynamical data) :	1
maxMarkovLag (for dynamical data):	1
maxMarkovLag (for dynamical data):	1
maxMarkovLag (for static data):	0
dbnMandatoryIdentityLags:	1

equivalentSampleSize:	1.0
maxParentCount:	5
maxTime	
10 Genes	60 Seconds
100 Genes	600 Seconds
minNetworkBeforeChecking:	1000

Appendix B

Pseudo-code: parallel implementation of reverse-engineering algorithm

{discretize local data expression profiles}
for $i = 1$ to $\frac{N}{p}$ **do**
    $localE\_dis(i,:) \leftarrow discretize(localE(i,:)$
    for $j = 1$ to $i - 1$ **do**
        {compute mutual information for each pair of local probes}
        $localMI(i,j) \leftarrow computeMI(localE\_dis(i,:), localE\_dis(j,:))$
    end for
end for
for $i = 1$ to $\frac{p}{2}$ **do**
    $\{P_{j(\bmod p)}$ sends its local discretized data expression profiles to $P_{(j+i)(\bmod p)}\}$
    $Send(localE\_dis, my\_rank + i)$
    $\{P_{(j+i)(\bmod p)}$ receives discretized data expression profiles from $P_{(j)(\bmod p)}\}$
    $E\_recv \leftarrow Recv(my\_rank - i)$
    {compute mutual information for all the pairs of probes where the first is the local probe while the second is the received probe}
    for $j = 1$ to $\frac{N}{p}$ **do**
        for $k = 1$ to $j - 1$ **do**
            $localMI(j,k) \leftarrow computeMI(localE\_dis(j,:), E\_recv(k,:))$
        end for
    end for
end for

Figure B.1: Parallel mutual information algorithm

Appendix C

Matlab codes

Listing C.1: Simulation of weighted adjacency matrix representing gene regulatory networks

```
1  function simulNet(nGenes, sparsity, Ntgts, Nregs)
2  global A     %%% A will contain the adjacency matricx representing
3               %%% the gene regulatory network (output)
4
5  %%% nGenes:     number of genes in the network
6  %%% sparsity:   average number of connection for each gene
7  %%% Ntgts:      number of non transcription factor genes
8  %%% Nregs:      number of transcription factor genes
9
10 NonzeroFrac = sparsity;   %%% Density of network
11
12 z = norminv(1-NonzeroFrac/2,0,1); %%% threshold
13
14 UR = randn(Ntgts,Nregs);      %%% generate a random matrix (will
15                               %%% be the up-right part of the final
16                               %%% matrix) from normal distribution
17
18 UR = UR.*(abs(UR)>z);    %%% keep only those values for which
19                          %%% the probability to be in the net
20                          %%% is greater than the threshold
21
22 LR = randn(Nregs);             %%% these lines of code allow to
23 UL = -eye(Ntgts);              %%% build a net in which some parts
24 LL = zeros(Nregs,Ntgts);      %%% of it are not randomized (these
```

```
25                                  %%% represent, respectively, the low-
26                                  %%% right, the upper-left and the low
27                                  %%% left part of the end matrix)
28
29  A = [UL UR ; LL LR];            %%% the final net is assembled
30                                  %%% from the parts created above
31
32  A = LR.*(LR>z) - eye(Nregs);    %%% keep only those values
33                                    %%% greater than the threshold
34                                    %%% and add negative values
35                                    %%% on the diagonal of it
36  end
```

Listing C.2: Simulation of static (steady-state) gene expressions level given a network topology expressed by the adjacency matrix output of the code reported in Listing C.1

```
1   function simulStaticExpr(nGenes, N, Ntgts, Nregs, policy)
2
3   global A Xss P  %%% these variables represent, respectively,
4                   %%% the input net, the matrix of simulated
5                   %%% experiments and the applied perturbation
6
7   %%% Output
8   %%% Xss:     matrix of simulated gene expression levels
9   %%% P:       perturbation matrix, perturbations applied
10  %%%          to the adjacency matrix, A, in order to
11  %%%          generate the gene expression levels
12
13
14  %%% Ntgts:   number of Non transcription factor genes
15  %%% Nregs:   number of Transcription factor genes
16  %%% N:       number of Experiments
17
18  if policy == 'global'
19      %%% global perturbation policy
20      %%% Create a random perturbation matrix
21      %%% from uniformly distributed values
22      P = [zeros(Ntgts,N);rand(Nregs,N)];
23  end
24
```

```
25 if policy == 'local'
26     %%% local perturbation policy
27     %%% Create a diagonal perturbation matrix
28     P = eye(nGenes,nGenes)];
29 end
30
31 Xss = inv(A)*(-P);  %%% Noise free data
32
33 return
```

Listing C.3: Generate and add Gaussian noise to the gene expression levels generated with the code in Listing C.2

```
1  function simulStaticExprNoisey(noiseLevel)
2
3  %%% Input
4  %%% noiseLevel: percentage of noise to add given the expression
5  %%%              level (mean of the Gaussian distribution)
6
7  global Xss nXss %%% these variables will contain, respectively,
8                  %%% simulated gene expressionlevels (input)
9                  %%% and the noisy gene expression levels
10
11 %%% level of the external noise
12 sX = noiseLevel;
13
14 %%% Noisy data points: the noise value is
15 %%% extracted from a normal distribution
16 %%% and than added to the input expressions
17 nXss = sX*randn(size(Xss))+Xss;
18 return
```

Listing C.4: Computes the Positive Predicted Value (PPV) and the Sensitivity given two adjacency matrix. One is the adjacency matrix representing the gene network as produced by the code in Listing C.1. The second is the adjacency matrix as obtained by the application of the reverse-engineering algorithms (Chapters 3 and 4) on the gene expression levels generated using the code in Listings C.2 and C.3

```
1  function [connections_predicted,ppv,sensitivity] = ppv_sensitivity_mod(AA,
       original_A)
```

```
2
3  %%% Code for computing the Positive Predicted Value (PPV) and the
4  %%% Sesitivity of the inference process of the reverse-engineering
5  %%% algorithms, which infer gene regulatory networks given gene
6  %%% gene expression profiles (and perturbations)
7
8  %%% Input
9  %%% AA:       adjacency matrix representing the gene regulatory
10 %%%          network as inferred by the reverse-engineering algorithms
11 %%%
12 %%% original_AA:     weighted adjacency matrix representing the real gene
13 %%%                  regulatory network to infer
14
15 %%% Output
16 %%% connections_predicted: number of connection predicted by the algorithm
17 %%% ppv:      positive predicted value, ratio between the number of correct
18 %%%          predictions and the total number of predictions
19 %%%
20 %%% sensitivity:     coverage of the real gene network, ratio between the
21 %%%                  number correct predictions and the total number of
22 %%%                  connections in the real network (original_AA)
23
24 connections_predicted = nnz(AA);
25
26 true_positives = nnz((AA+(original_A~=0))==2);
27 false_positives = nnz(((original_A~=0)-AA)==-1);
28 false_negatives = nnz((AA-(original_A~=0))==-1);
29
30 ppv = true_positives/(true_positives + false_positives);
31 sensitivity = true_positives/(true_positives + false_negatives);
```

Listing C.5: Gene Ontology Enrichment Analysis, computation of the enrichment (p-value) of a Gene Ontology term in a set of genes

```
1  function [p_vals, idxs] = computeGOEnrichment(genes,mat)
2
3  %%% Input
4  %%% genes:  set of genes
5  %%% mat:    gene ontology terms associated to all the gene of the
6  %%%         mammalian system considered (human or mouse)
7
8  %%% Output
```

```
%%% p_vals: list of enrichment score (p-values) of the gene
%%%          ontology terms associated to the genes in input
%%%
%%% idxs:   Gene Ontology codes associates to the GO terms

% number of genes in the mammalian system with a given GO term
m_vet = sum(mat);

% number of times a GO term is associated to
% a gene of the set of genes in input
k_vet = sum(mat(genes,:));

idxs = find(k_vet); % indexes of non null elements
N = nnz(sum(mat,2)); % # of genes in the mammalia system

t_vet = sum(mat,2)~=0;
n = nnz(t_vet(genes)); % number of genes of the mammalian system

k_vet = full(k_vet(idxs)); m_vet = full(m_vet(idxs));

% compute the cumulative hypergeometric distribution
% as sum of the hypergeometric probabilities
p_vals=[];
for kk=1:numel(k_vet)
    p_vals(kk)= sum(hygepdf(k_vet(kk):min(m_vet(kk),n), N, m_vet(kk), n));
end
```

Listing C.6: Discretises a vector of real values (gene expression levels) into a predetermined numbers of integer values (or bins)

```
function A_dis = discretizer(A,C)

%%% Input
%%% A:       vector containing real values
%%% C:       number of discretised states

%%% Output
%%% A_dis:   vector containing the discretised values

%get the number of rows and columns of A
[n,m] = size(A);

```

```
% create a boundary array
perc = 0:1/C:1;
perc = perc(2:C);

% find the bounds of the intervals where to discreise the values of A
bnd = quantile(reshape(A,1,n*m),perc);

% collect the data into bins
 A_dis = A<bnd(1);
for i=2:C-1
    A_dis = A_dis + i*(A<bnd(i) & A>=bnd(i-1));
end
A_dis = A_dis + C*(A>=bnd(C-1));
```

Listing C.7: Computes the Mutual Information given a pair of discretised vectors obtained by the application of the code in Listing C.6

```
function MI2 = computeMI(A,B,nbins)

%%% Input
%%% A,B:    vectors containing discretised values
%%% nbins:  number of discritised states

%%% Output
%%% MI2 Mutual Information beteen vector A and B

% compute the frequencies of the outcomes
p = findStatistics(A',B',nbins);

% compute the Mutual Inforamtion
MI2 = computePairMI(p,nbins);


%% compute the frequencies of the outcomes
function p = findStatistics(A,B,nbins)

% account for two pair of genes
vet = [A B];
for h=1:nbins
    for k=1:nbins
        %account for a pair of values
        p(h,k) = sum(ismember(vet,[h,k],'rows'));
```

```
26      end
27  end
28
29  p=p/sum(sum(p)); %frequencies from counts
30
31
32  %% compute Mutual Information giveh the parameter vector pi
33  function MI2 = computePairMI(p)
34
35  t_p = sum(p,2);
36  a = -sum(t_p.*log2(t_p+(t_p==0)));
37
38  t_p = sum(p,1);
39  b = -sum(t_p.*log2(t_p+(t_p==0)));
40
41  ab = -sum(sum(p.*log2(p+(p==0))));
42
43  MI2 = a+b-ab;
```

Bibliography

[1] T Akutsu, S Miyano, and S Kuhara. Identification of genetic networks from a small number of gene expression patterns under the Boolean network model. *Pac Symp Biocomput*, pages 17–28, 1999.

[2] T Akutsu, S Miyano, and S Kuhara. Inferring qualitative relations in genetic networks and metabolic pathways. *Bioinformatics*, 16(8):727–734, Aug 2000.

[3] Reka Albert and Albert-Laszlo Barabasi. Statistical mechanics of complex networks. *Topology*, page 54, 2001.

[4] Andrey Alexeyenko and Erik L. Sonnhammer. Global networks of functional coupling in eukaryotes from comprehensive data integration. *Genome research*, 19(6):1107–1116, June 2009.

[5] A Ambesi-Impiombato and D di Bernardo. Computational biology and drug discovery: From single-target to network drugs. *Current Bioinformatics*, 1:3–13, 2006.

[6] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock. Gene ontology: tool for the unification of biology. the gene ontology consortium. *Nature genetics*, 25(1):25–29, May 2000.

[7] Matt Baker, Ian R. Mackenzie, Stuart M. Pickering-Brown, Jennifer Gass, Rosa Rademakers, Caroline Lindholm, Julie Snowden, Jennifer Adamson, A. Dessa Sadovnick, Sara Rollinson, Ashley Cannon, Emily Dwosh, David Neary, Stacey Melquist, Anna Richardson, Dennis Dickson, Zdenek Berger, Jason Eriksen, Todd Robinson, Cynthia Zehr, Chad A. Dickey, Richard Crook, Eileen McGowan, David Mann, Bradley Boeve, Howard Feldman, and Mike Hutton. Mutations in progranulin cause tau-negative frontotemporal dementia linked to chromosome 17. *Nature*, 442(7105):916–919, July 2006.

[8] Sourav Bandyopadhyay, Roded Sharan, and Trey Ideker. Systematic identification of functional orthologs based on protein network comparison. *Genome research*, 16(3):428–35, 2006.

[9] Mukesh Bansal, Vincenzo Belcastro, Alberto Ambesi-Impiombato, and Diego di Bernardo. How to infer gene networks from expression profiles. *Molecular systems biology*, 3, February 2007.

[10] Mukesh Bansal, Giusy Della Gatta, and Diego di Bernardo. Inference of gene regulatory networks and compound mode of action from time course gene expression profiles. *Bioinformatics*, 22(7):815–822, Apr 2006.

[11] AL Barabasi and E. Bonabeau. Scale-free networks. *Scientific American*, 288(5):509, 2003.

[12] Katia Basso, Adam A Margolin, Gustavo Stolovitzky, Ulf Klein, Riccardo Dalla-Favera, and Andrea Califano. Reverse engineering of regulatory networks in human B cells. *Nat Genet*, 37(4):382–390, Apr 2005.

[13] M A Beer and S Tavazoie. Predicting gene expression from sequence. *Cell*, pages 185–198, 2004.

[14] vincenzo Belcastro, Francesco Gregoretti, Gennaro Oliva, and Diego di Bernardo. Parallel computing algorithms for reverse-engineering and analysis of genome-wide gene regulatory networks from gene expression profiles. *Manuscript in preparation.*

[15] vincenzo Belcastro, Velia Siciliano, Francesco Gregoretti, Pratibha Mithbaokar, Francesco Iorio, Gennaro Oliva, Nicola Brunetti-Pierri, and Diego di Bernardo. A comprehensive gene regulatory network predicts transcriptome organization and gene function. *Submitted.*

[16] Bernard. *Density Estimation for Statistics and Data Analysis (Chapman & Hall/CRC Monographs on Statistics & Applied Probability).* Chapman and Hall/CRC, 1 edition, April 1986.

[17] Richard Bonneau, David J Reiss, Paul Shannon, Marc Facciotti, Leroy Hood, Nitin S Baliga, and Vesteinn Thorsson. The Inferelator: an algorithm for learning parsimonious regulatory networks from systems-biology data sets de novo. *Genome Biol*, 7(5):R36, 2006. Evaluation Studies.

[18] Alice Bossi and Ben Lehner. Tissue specificity and the human protein interaction network. *Mol Syst Biol*, 5, April 2009.

[19] P Brazhnik, A de la Fuente, and P Mendes. Artificial gene networks for objective comparison of analysis algorithms. *Bioinformatics*, 19 Suppl 2:II122–II129, 2003.

[20] Alvis Brazma, Pascal Hingamp, John Quackenbush, Gavin Sherlock, Paul Spellman, Chris Stoeckert, John Aach, Wilhelm Ansorge, Catherine A. Ball, Helen C. Causton, Terry Gaasterland, Patrick Glenisson, Frank C. P. Holstege, Irene F. Kim, Victor Markowitz, John C. Matese, Helen Parkinson, Alan Robinson, Ugis Sarkans, Steffen Schulze-Kremer, Jason Stewart, Ronald Tay-

lor, Jaak Vilo, and Martin Vingron. Minimum information about a microarray experiment (miame)-toward standards for microarray data. *Nat Genet*, 29(4):365–371, December 2001.

[21] A. J. Butte and I. S. Kohane. Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. In *Pacific Symposium of Biocomputing*, pages 418–429, Children's Hospital Informatics Program, Boston, MA 02115, USA., 2000.

[22] Mafalda Cacciottolo, Vincenzo Belcastro, Steve Laval, Katie Bushby, Diego di Bernardo, and Vincenzo Nigro. Reverse-engineering gene network identifies new dysferlin interacting proteins. *J Biol Chem*, November 2010.

[23] Irene Cantone, Lucia Marucci, Francesco Iorio, Maria Aurelia Ricci, Vincenzo Belcastro, Mukesh Bansal, Stefania Santini, Mario di Bernardo, Diego di Bernardo, and Maria Pia Cosma. A yeast synthetic network for in vivo assessment of reverse-engineering and modeling approaches. *Cell*, 137(1):172–81, April 2009.

[24] Maria S. Carro, Wei K. Lim, Mariano J. Alvarez, Robert J. Bollo, Xudong Zhao, Evan Y. Snyder, Erik P. Sulman, Sandrine L. Anne, Fiona Doetsch, Howard Colman, Anna Lasorella, Ken Aldape, Andrea Califano, and Antonio Iavarone. The transcriptional network for mesenchymal transformation of brain tumours. *Nature*, 463(7279):318–325, December 2009.

[25] A M Cataldo, P A Paskevich, E Kominami, and R A Nixon. Lysosomal hydrolases of different classes are abnormally distributed in brains of patients with Alzheimer disease. *Proceedings of the National Academy of Sciences of the United States of America*, 88(24):10998–11002, 1991.

[26] T Chen, H L He, and G M Church. Modeling gene expression with differential equations. *Pac Symp Biocomput*, pages 29–40, 1999.

[27] Marc Cruts, Ilse Gijselinck, Julie van der Zee, Sebastiaan Engelborghs, Hans Wils, Daniel Pirici, Rosa Rademakers, Rik Vandenberghe, Bart Dermaut, Jean-Jacques Martin, Cornelia van Duijn, Karin Peeters, Raf Sciot, Patrick Santens, Tim De Pooter, Maria Mattheijssens, Marleen Van den Broeck, Ivy Cuijt, Krist'l Vennekens, Peter P. De Deyn, Samir Kumar-Singh, and Christine Van Broeckhoven. Null mutations in progranulin cause ubiquitin-positive frontotemporal dementia linked to chromosome 17q21. *Nature*, 442(7105):920–924, July 2006.

[28] H de Jong. Modeling and simulation of genetic regulatory systems: A literature review. *J. Comp. Biol.*, 9:67–103, 2002.

[29] H. de Jong. Genetic Network Analyzer: qualitative simulation of genetic regulatory networks. *Bioinformatics*, 19(3):336–344, 2003.

[30] Alberto de la Fuente, Nan Bing, Ina Hoeschele, and Pedro Mendes. Discovery of meaningful associations in genomic data using partial correlation coefficients. *Bioinformatics*, 20(18):3565–3574, Dec 2004. Evaluation Studies.

[31] G. Della Gatta, M. Bansal, A. Ambesi-Impiombato, D. Antonini, C. Missero, and di Bernardo D. Direct targets of the trp63 transcription factor revealed by a combination of gene expression profiling and reverse engineering. *Genome Res.*, 6(18):939–948, Jun 2008.

[32] P D'Haeseleer, X Wen, S Fuhrman, and R Somogi. Linear modeling of mRNA expression levels during CNS development and injury. *Proc. Pacific Symp. Biocomp.*, 4:41–52, 1999.

[33] D di Bernardo, MJ Thompson, TS Gardner, SE Chobot, EL Eastwood, AP Wojtovich, SJ Elliott, SE Schaus, and JJ Collins. Chemogenomic profiling on a genome-wide scale using reverse-engineered gene networks. *Nature Biotechnology*, 23:377–383, 2005.

[34] Norbert Dojer, Anna Gambin, Andrzej Mizera, Bartek Wilczynski, and Jerzy Tiuryn. Applying dynamic Bayesian networks to perturbed gene expression data. *BMC Bioinformatics*, 7:249, 2006.

[35] Ron Edgar, Michael Domrachev, and Alex E. Lash. Gene expression omnibus: Ncbi gene expression and hybridization array data repository. *Nucl. Acids Res.*, 30(1):207–210, January 2002.

[36] M Eisen, PT Spellman, PO Brown, and D Botstein. Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci. USA*, page 1486314868, 1998.

[37] Jason L Eriksen and Ian R a Mackenzie. Progranulin: normal function and role in neurodegeneration. *Journal of neurochemistry*, 104(2):287–97, January 2008.

[38] J Faith and TS Gardner. Reverse-engineering transcription control networks. *Physics of Life Reviews*, 2:65–88, 2005.

[39] Jeremiah J Faith, Boris Hayete, Joshua T Thaden, Ilaria Mogno, Jamey Wierzbowski, Guillaume Cottarel, Simon Kasif, James J Collins, and Timothy S Gardner. Large-scale mapping and validation of Escherichia coli transcriptional regulation from a compendium of expression profiles. *PLoS Biol*, 5(1):e8, Jan 2007.

[40] Jeremiah J. Faith, Boris Hayete, Joshua T. Thaden, Ilaria Mogno, Jamey Wierzbowski, Guillaume Cottarel, Simon Kasif, James J. Collins, and Timothy S. Gardner. Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles. *PLoS Biology*, 5(1):e8+, January 2007.

[41] BC Foat, AV Morozov, and H J Bussemaker. Statistical mechanical modeling of genome-wide transcription factor occupancy data by matrixreduce. *Bioinformatics*, 22:e141–e149, 2006.

[42] Brendan J J. Frey and Delbert Dueck. Clustering by passing messages between data points. *Science*, 315, January 2007.

[43] N Friedman, M Linial, I Nachman, and D Pe'er. Using Bayesian networks to analyze expression data. *J Comput Biol*, 7(3-4):601–620, 2000.

[44] TS Gardner, D di Bernardo, D Lorenz, and JJ Collins. Inferring genetic networks and identifying compound mode of action via expression profiling. *Science*, 301:102–105, 2003.

[45] Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin. *Bayesian Data Analysis, Second Edition.* Chapman & Hall/CRC, July 2003.

[46] B. Goebel, Z. Dawy, J. Hagenauer, and J.C. Mueller. An approximation to the distribution of finite sample size mutual information estimates. *Communications, 2005. ICC 2005. 2005 IEEE International Conference on*, 2:1102–1106, May 2005.

[47] Francesco Gregoretti, Vincenzo Belcastro, Diego di Bernardo, and Gennaro Oliva. A parallel implementation of the network identification by multiple regression (nir) algorithm to reverse-engineer regulatory gene networks. *PloS one*, 5(4), 2010.

[48] M Gustafsson, M Hrnquist, J Lundstrm, J Bjrkegren, and J Tegnr. Reverse Engineering of Gene Networks with LASSO and Nonlinear Basis Functions. *Annals of the New York Academy of Sciences*, 1158:265–275, 2009.

[49] Gerald J. Hahn. and Samuel S. Shapiro. *Statistical models in engineering.* Wiley & Sons, New York, 1994.

[50] A J Hartemink, D K Gifford, T S Jaakkola, and R A Young. Combining location and expression data for principled discovery of genetic regulatory network. *Proc. Pacific Symp. Biocomp.*, 7:437–449, 2002.

[51] Zhiheng He, Colin H P Ong, Jaroslava Halper, and Andrew Bateman. Progranulin is a mediator of the wound response. *Nature medicine*, 9(2):225–9, February 2003.

[52] Amanda Helip-Wooley and Jess G. Thoene. Sucrose-induced vacuolation results in increased expression of cholesterol biosynthesis and lysosomal genes. *Experimental Cell Research*, 292(1):89 – 100, 2004.

[53] Graham J. Hickman and T. Charlie Hodgman. Inference of gene regulatory networks using boolean-network inference methods. *Journal of bioinformatics and computational biology*, 7(6):1013–1029, December 2009.

[54] Antti Honkela, Charles Girardot, E. Hilary Gustafson, Ya-Hsin H. Liu, Eileen E. Furlong, Neil D. Lawrence, and Magnus Rattray. Model-based method for transcription factor target identification with limited data. *Proceedings of the National Academy of Sciences of the United States of America*, 107(17):7793–7798, April 2010.

[55] T R Hughes et al. Functional discovery via a compendium of expression profiles. *Cell*, 102:109–126, 2000.

[56] Marcus Hutter. Distribution of mutual information. *Advanced in Neuronal Information Processing Systems*, 18:339–406, 2004.

[57] Francesco Iorio, Roberta Bosotti, Emanuela Scacheri, Vincenzo Belcastro, Pratibha Mithbaokar, Rosa Ferriero, Loredana Murino, Roberto Tagliaferri, Nicola Brunetti-Pierri, Antonella Isacchi, and Diego di Bernardo. Discovery of drug mode of action and drug repositioning from transcriptional responses.

Proceedings of the National Academy of Sciences of the United States of America, pages 1–6, August 2010.

[58] Ivan Ivanov. Boolean models of genomic regulatory networks: reduction mappings, inference, and external control. *Current genomics*, 10(6):375–87, September 2009.

[59] Victoria L Johnson, Maria I F Scott, Sarah V Holt, Deema Hussein, and Stephen S Taylor. Bub1 is required for kinetochore localization of BubR1, Cenp-E, Cenp-F and Mad2, and chromosome congression. *Journal of cell science*, 117(Pt 8):1577–89, 2004.

[60] G. Joshi-Tope, M. Gillespie, I. Vastrik, P. D'Eustachio, E. Schmidt, B. de Bono, B. Jassal, G. R. Gopinath, G. R. Wu, L. Matthews, S. Lewis, E. Birney, and L. Stein. Reactome: a knowledgebase of biological pathways. *Nucleic Acids Res*, 33(Database issue), January 2005.

[61] Akira Kajiki, Louis H Liu, and Peggy J Pula. Sources of Extracellular Lysosomal Enzymes Released by Developing and Healing Inflammatory Lesions in. *Journal of Leukocyte Biology*, 116:104–116, 1988.

[62] Litsa E. Karageorgos, Elizabeth L. Isaac, Doug A. Brooks, Elaine M. Ravenscroft, Richard Davey, John J. Hopwood, and Peter J. Meikle. Lysosomal biogenesis in lysosomal storage disorders. *Experimental Cell Research*, 234(1):85 - 97, 1997.

[63] A. E. Kel, E. Gössling, I. Reuter, E. Cheremushkin, O. V. Kel-Margoulis, and E. Wingender. Match: A tool for searching transcription factor binding sites in dna sequences. *Nucleic acids research*, 31(13):3576–3579, July 2003.

[64] Hans H Klünemann, Milan Elleder, Wolfgang E. Kaminski, Karen Snow, Janis M. Peyser, John F. O'Brien, David Munoz, Gerd Schmitz, Helmfried E.

Klein, and William W. Pendlebury. Frontal lobe atrophy due to a mutation in the cholesterol binding protein hel/npc2. *Annals of Neurology*, 52(6):743–749, 2002.

[65] Katrin Kollmann, Kudzai E Mutenda, Martina Balleininger, Ellen Eckermann, Kurt von Figura, Bernhard Schmidt, and Torben Lübke. Identification of novel lysosomal matrix proteins by proteome analysis. *Proteomics*, 5(15):3966–78, October 2005.

[66] G. N. Lance and W. T. Williams. A general theory of classificatory sorting strategies: 1. hierarchical systems. *The Computer Journal*, 9(4):373–380, February 1967.

[67] Tong Ihn Lee, Nicola J Rinaldi, Francois Robert, Duncan T Odom, Ziv Bar-Joseph, Georg K Gerber, Nancy M Hannett, Christopher T Harbison, Craig M Thompson, Itamar Simon, Julia Zeitlinger, Ezra G Jennings, Heather L Murray, D Benjamin Gordon, Bing Ren, John J Wyrick, Jean-Bosco Tagne, Thomas L Volkert, Ernest Fraenkel, David K Gifford, and Richard A Young. Transcriptional regulatory networks in Saccharomyces cerevisiae. *Science*, 298(5594):799–804, Oct 2002.

[68] S Liang, S Fuhrman, and R Somogyi. REVEAL, a general reverse engienering algorithm for inference of genetic network architectures. *Proc. Pacific Symp. Biocomp.*, 3:18–29, 1998.

[69] Erez Lieberman-Aiden, Nynke L. van Berkum, Louise Williams, Maxim Imakaev, Tobias Ragoczy, Agnes Telling, Ido Amit, Bryan R. Lajoie, Peter J. Sabo, Michael O. Dorschner, Richard Sandstrom, Bradley Bernstein, M. A. Bender, Mark Groudine, Andreas Gnirke, John Stamatoyannopoulos, Leonid A. Mirny, Eric S. Lander, and Job Dekker. Comprehensive mapping of

long-range interactions reveals folding principles of the human genome. *Science*, 326(5950):289–293, October 2009.

[70] R. Linding, R. B. Russell, V. Neduva, and T. J. Gibson. Globplot: Exploring protein sequences for globularity and disorder. *Nucleic acids research*, 31(13):3701–3708, July 2003.

[71] Huan Liu, Farhad Hussain, Chew L. Tan, and Manoranjan Dash. Discretization: An enabling technique. *Data Mining and Knowledge Discovery*, 6(4):393–423, October 2002.

[72] L Ljung. *System Identification: Theory for the User*. Prentice Hall, Upper Saddle River, NJ, 1999.

[73] Lennart Ljung. *System Identification: Theory for the User (2nd Edition)*. Prentice Hall PTR, December 1998.

[74] Amity L Manning, Neil J Ganem, Samuel F Bakhoum, Michael Wagenbach, Linda Wordeman, and Duane A Compton. The kinesin-13 proteins Kif2a, Kif2b, and Kif2c/MCAK have distinct roles during mitosis in human cells. *Molecular Biology of the Cell*, 18(August):2970 –2979, 2007.

[75] Daniel Marbach, Robert J Prill, Thomas Schaffter, Claudio Mattiussi, Dario Floreano, and Gustavo Stolovitzky. Revealing strengths and weaknesses of methods for gene network inference. *Proceedings of the National Academy of Sciences of the United States of America*, 107(14):6286–91, 2010.

[76] A Margolin, I Nemenman, K Basso, C Wiggins, G Stolovitzky, R Della Favera, and A Califano. Aracne: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. *BMC Bioinformatics*, pages S1 (arXiv: q–bio.MN/0410037), 2006.

[77] Adam A Margolin, Ilya Nemenman, Katia Basso, Chris Wiggins, Gustavo Stolovitzky, Riccardo Dalla Favera, and Andrea Califano. ARACNE: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. *BMC Bioinformatics*, 7 Suppl 1:S7, 2006.

[78] J Meldolesi. Surface wound healing: a new, general function of eukaryotic cells. *Journal of cellular and molecular medicine*, 7(3):197–203, 2003.

[79] T. Minka. Estimating a dirichlet distribution, 2003.

[80] I Nachman, A Regev, and N Friedman. Inferring quantitative models of regulatory networks from expression data. *Bioinformatics*, 20 Suppl 1:248–256, Aug 2004.

[81] Swati Naphade, Kristina Kigerl, Lyn Jakeman, Sandra Kostyk, Phillip Popovich, and Jeff Kuret. Progranulin expression is upregulated after spinal contusion in mice. *Acta Neuropathologica*, 119, November 2009.

[82] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45:167–256, 2003.

[83] MEJ Newman. The structure and function of complex networks. *Arxiv preprint cond-mat/0303516*, 45:167–256, 2003.

[84] T. Obayashi, S. Hayashi, M. Shibaoka, M. Saeki, H. Ohta, and K. Kinoshita. Coxpresdb: a database of coexpressed gene networks in mammals. *Nucleic Acids Res*, 36(Database issue), January 2008.

[85] Kazuhiro Ohmi, David S. Greenberg, Kavitha S. Rajavel, Sergey Ryazantsev, Hong Hua Li, and Elizabeth F. Neufeld. Activated microglia in cortex of mouse models of mucopolysaccharidoses I and IIIB. *Proceedings of the National Academy of Sciences of the United States of America*, 100(4):1902–1907, 2003.

[86] A. Oliva, A. Rosebrock, F. Ferrezuelo, S. Pyne, H. Chen, S. Skiena, B. Futcher, and J. Leatherwood. The cell cycle-regulated genes of schizosaccharomyces pombe. *PLoS Biol*, 3(7), July 2005.

[87] H. Parkinson, M. Kapushesky, M. Shojatalab, N. Abeygunawardena, R. Coulson, A. Farne, E. Holloway, N. Kolesnykov, P. Lilja, M. Lukk, R. Mani, T. Rayner, A. Sharma, E. William, U. Sarkans, and A. Brazma. Arrayexpress–a public database of microarray experiments and gene expression profiles. *Nucleic Acids Res*, 35:D747–D750, Jan 2007.

[88] H. Parkinson, M. Kapushesky, M. Shojatalab, N. Abeygunawardena, R. Coulson, A. Farne, E. Holloway, N. Kolesnykov, P. Lilja, M. Lukk, R. Mani, T. Rayner, A. Sharma, E. William, U. Sarkans, and A. Brazma. Arrayexpress–a public database of microarray experiments and gene expression profiles. *Nucleic Acids Res*, 35(Database issue), January 2007.

[89] Heidi S. Phillips, Samir Kharbanda, Ruihuan Chen, William F. Forrest, Robert H. Soriano, Thomas D. Wu, Anjan Misra, Janice M. Nigro, Howard Colman, and Liliana Soroceanu. Molecular subclasses of high-grade glioma predict prognosis, delineate a pattern of disease progression, and resemble stages in neurogenesis. *Cancer Cell*, 9(3):157–173, March 2006.

[90] A Prakash and M Tompa. Discovery of regulatory elements in vertebrates through comparative genomics. *Nat Biotechnol*, pages 1249–1256, 2005.

[91] Robert J Prill, Daniel Marbach, Julio Saez-Rodriguez, Peter K Sorger, Leonidas G Alexopoulos, Xiaowei Xue, Neil D Clarke, Gregoire Altan-Bonnet, and Gustavo Stolovitzky. Towards a rigorous assessment of systems biology models: the DREAM3 challenges. *PloS one*, 5(2):e9202, 2010.

[92] Tim Raemaekers, Katharina Ribbeck, Joel Beaudouin, Wim Annaert, Mark Van Camp, Ingrid Stockmans, Nico Smets, Roger Bouillon, Jan Ellenberg, and

Geert Carmeliet. NuSAP, a novel microtubule-associated protein involved in mitotic spindle organization. *The Journal of cell biology*, 162(6):1017–29, 2003.

[93] Timothy Ravasi, Harukazu Suzuki, Carlo Vittorio Cannistraci, Shintaro Katayama, Vladimir B Bajic, Kai Tan, Altuna Akalin, Sebastian Schmeier, Mutsumi Kanamori-Katayama, Nicolas Bertin, Piero Carninci, Carsten O Daub, Alistair R R Forrest, Julian Gough, Sean Grimmond, Jung-Hoon Han, Takehiro Hashimoto, Winston Hide, Oliver Hofmann, Hideya Kawaji, Atsutaka Kubosaki, Timo Lassmann, Erik van Nimwegen, Chihiro Ogawa, Rohan D Teasdale, Jesper Tegnér, Boris Lenhard, Sarah a Teichmann, Takahiro Arakawa, Noriko Ninomiya, Kayoko Murakami, Michihira Tagami, Shiro Fukuda, Kengo Imamura, Chikatoshi Kai, Ryoko Ishihara, Yayoi Kitazume, Jun Kawai, David a Hume, Trey Ideker, and Yoshihide Hayashizaki. An atlas of combinatorial transcriptional regulation in mouse and man. *Cell*, 140(5):744–52, 2010.

[94] E. Ravasz, A. L. Somera, D. A. Mongru, Z. N. Oltvai, and A. L. Barabasi. Hierarchical organization of modularity in metabolic networks. *Science*, 297(5586):1551–1555, August 2002.

[95] G. Rustici, J. Mata, K. Kivinen, P. Lio, C.J. Penkett, G. Burns, J. Hayles, A. Brazma, P. Nurse, and J. Bahler. Periodic gene expression program of the fission yeast cell cycle. *Nat Genet*, 36(8):809–817, August 2004.

[96] Marco Sardiello, Michela Palmieri, Alberto di Ronza, Diego Luis Medina, Marta Valenza, Vincenzo Alessandro Gennarino, Chiara Di Malta, Francesca Donaudy, Valerio Embrione, Roman S Polishchuk, Sandro Banfi, Giancarlo Parenti, Elena Cattaneo, and Andrea Ballabio. A gene network regulating lysosomal biogenesis and function. *Science (New York, N.Y.)*, 325(5939):473–7, 2009.

[97] E Segal, B Taskar, A Gasch, N Friedman, and D Koller. Rich probabilistic models for gene expression. *Bioinformatics*, 17 Suppl 1:243–252, 2001.

[98] Eran Segal, Michael Shapira, Aviv Regev, Dana Pe'er, David Botstein, Daphne Koller, and Nir Friedman. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nat Genet*, 34(2):166–176, Jun 2003.

[99] Roded Sharan, Silpa Suthram, Ryan M Kelley, Tanja Kuhn, Scott McCuine, Peter Uetz, Taylor Sittler, Richard M Karp, and Trey Ideker. Conserved patterns of protein interaction in multiple species. *Proceedings of the National Academy of Sciences of the United States of America*, 102(6):1974–9, February 2005.

[100] V Anne Smith, Erich D Jarvis, and Alexander J Hartemink. Evaluating functional network inference using simulations of complex biological systems. *Bioinformatics*, 18 Suppl 1:216–224, 2002. Evaluation Studies.

[101] V. Anne Smith, Erich D. Jarvis, and Alexander J. Hartemink. Evaluating functional network inference using simulations of complex biological systems. *Bioinformatics*, 18(suppl_1):S216–224, July 2002.

[102] V Anne Smith, Erich D Jarvis, and Alexander J Hartemink. Influence of network topology and data collection on network inference. *Pac Symp Biocomput*, pages 164–175, 2003.

[103] R. R. Sokal and C. D. Michener. A statistical method for evaluating systematic relationships. *University of Kansas Scientific Bulletin*, 28:1409–1438, 1958.

[104] R. Steuer, J. Kurths, C. O. Daub, J. Weise, and J. Selbig. The mutual information: Detecting and evaluating dependencies between variables. *Bioinformatics*, 18(suppl_2):S231–240, October 2002.

[105] R Steuer, J Kurths, C O Daub, J Weise, and J Selbig. The mutual information: detecting and evaluating dependencies between variables. *Bioinformatics*, 18 Suppl 2:231–240, 2002. Evaluation Studies.

[106] Joshua M. Stuart, Eran Segal, Daphne Koller, and Stuart K. Kim. A gene-coexpression network for global discovery of conserved genetic modules. *Science*, 302(5643):249–255, October 2003.

[107] MG Tadesse, M Vannucci, and P Lio. Identification of dna regulatory motifs using bayesian variable selection. *Bioinformatics*, pages 2556–2561, 2004.

[108] J Tegner, M K Yeung, J Hasty, and Collins J J. Reverse engineering gene networks: integrating genetic perturbations with dynamical modeling. *Proc Natl Acad Sci U S A*, 100(10):5944–5949, May 2003.

[109] L Turchi, M Fareh, E Aberdam, S Kitajima, F Simpson, C Wicking, D Aberdam, and T Virolle. ATF3 and p15PAF are novel gatekeepers of genomic integrity upon UV stress. *Cell death and differentiation*, 16(5):728–37, 2009.

[110] Tim Van den Bulcke, Koenraad Van Leemput, Bart Naudts, Piet van Remortel, Hongwu Ma, Alain Verschoren, Bart De Moor, and Kathleen Marchal. SynTReN: a generator of synthetic gene expression data for design and analysis of structure learning algorithms. *BMC bioinformatics*, 7:43, 2006.

[111] E P van Someren, B L T Vaes, W T Steegenga, A M Sijbers, K J Dechering, and M J T Reinders. Least absolute regression network analysis of the murine osteoblast differentiation network. *Bioinformatics*, 22(4):477–484, Feb 2006.

[112] E P van Someren, L F Wessels, and M J Reinders. Linear modeling of genetic networks from experimental data. *Proc Int Conf Intell Syst Mol Biol*, 8:355–366, 2000.

[113] Tanya Vavouri, Jennifer I. Semple, Rosa Garcia-Verdugo, and Ben Lehner. Intrinsic protein disorder and interaction promiscuity are widely associated with dosage sensitivity. *Cell*, 138(1):198–208, July 2009.

[114] D C Weaver, C T Workman, and G D Stormo. Modeling regulatory networks with weight matrices. *Pac Symp Biocomput*, pages 112–123, 1999.

[115] Xuebing Wu, Rui Jiang, Michael Q. Zhang, and Shao Li. Network-based global inference of human disease genes. *Mol Syst Biol*, 4, May 2008.

[116] Zhi Xu, William Farver, Sarala Kodukula, and Judith Storch. Regulation of sterol transport between membranes and npc2. *Biochemistry*, 47(42):11134–11143, 2008.

[117] Tetsumori Yamashima, Yukihiko Kohda, Katsuhiro Tsuchiya, Takashi Ueno, Junkoh Yamashita, and Tohru Yoshioka. Inhibition of ischaemic hippocampal neuronal death in primates with cathepsin B inhibitor CA-074 : a novel strategy for neuroprotection based on calpain cathepsin hypothesis . *Neuroscience*, 10(November 1997):1723–1733, 1998.

[118] M K S Yeung, J Tegnér, and J J Collins. Reverse engineering gene networks using singular value decomposition and robust regression. *Proc. Natl. Acad. Sci. U.S.A.*, 99:6163–6168, 2002.

[119] Jing Yu, V. Anne Smith, Paul P. Wang, Alexander J. Hartemink, and Erich D. Jarvis. Advances to bayesian network inference for generating causal networks from observational biological data. *Bioinformatics*, 20(18):3594–3603, 2004.

[120] Min Zou and Suzanne D Conzen. A new dynamic Bayesian network (DBN) approach for identifying gene regulatory networks from time course microarray data. *Bioinformatics*, 21(1):71–79, Jan 2005. Comparative Study.